Khepera®

USER MANUAL

Version 5.02

K-Team

Lausanne, 12 March 1999

Documentation author

K-Team
Ch. de Vuasset, CP 111
1028 Préverenges
Switzerland

email: info@k-team.com
WWW: http://www.k-team.com

Trademark Acknowledgements

NOTICE:

- The contents of this manual are subject to change without notice.
- All efforts have been made to ensure the accuracy of the content of this manual. However, should any error be detected, please inform K-Team.
- The above notwithstanding, K-Team can assume no responsibility for any error in this manual.

TABLE OF CONTENT

1 INTRODUCTION

Khepera has originally been designed as a research and teaching tool in the framework of a Swiss Research Priority Program. It allows confrontation to the real world of algorithms developed in simulation for trajectory execution, obstacle avoidance, preprocessing of sensory information, hypothesis on behaviours processing. To be able to program the robot easily, LabVIEW® is proposed as a development environment. It is a graphical programming software, basically dedicated to instrumentation, which allows quick development of input-output interfaces, a necessity when dealing with the real world, by definition unpredictable and noisy. Please note that LabVIEW® is just a suggestion and is absolutely not needed to use the robot. Any other environment able to deal with the serial port of your computer can be use instead of LabVIEW®. The communication protocol implemented on Khepera and used by LabVIEW® is presented in chapter 6 and described in detail in appendix A.

1.1 How to Use this Manual

This manual is organised into six chapters and one appendix. To learn how to make the best use of your Khepera robot you are urged to read all of chapters 1 through 5. Chapter 6 presents the serial communication protocol that makes a remote control from a workstation possible. You need to read the chapter 7 if you use the software LabVIEW®. The appendix can be referred to as necessary.

Chapter 1	Gives an introduction to this manual and the Khepera robot.
Chapter 2	explains the contents of the package.
Chapter 3	explains the functionality of every item present in the package.
Chapter 4	explains how to make the first test of the robot after unpacking.
Chapter 5	gives the standard working configurations.
Chapter 6	presents the serial communication protocol.
Chapter 7	is addressed to users of LabVIEW®. It shows simple virtual instruments (VI) to control the robot functionality and a little example of programming in this environment.
Appendix A	details the commands of the serial communication protocol.
Appendix B	details the connectors pinning.
Appendix C	details how to open the robot, which means disconnect the upper from the lower part, and change the ROM. Both these operations has to be made only if really necessary!
Appendix D	gives some detail on the configuration of some common terminal emulators.
Appendix E	details the different running modes.

1.2 Safety Precautions

Check the unit's operating voltage before operation.
It must be identical with that of your local power supply. The operating voltage is indicated on the nameplate at the rear of the power supply.

Don't plug or unplug any connector when the system is switched ON.
All connections (including extension addition or disconnection) must be made when the robot and the interface are switched OFF. Otherwise damages can occur.

Switch OFF the robot if you will not use it for more than a day.
Disconnect the power supply removing it from the wall socket.

Do not open the robot (separate upper from lower part) if you do not have been explicitly allowed.
Disconnect the CPU to the sensory-motor board only if you have been allowed by a specific documentation. Perform this operation following carefully the instructions given in appendix C.

Do not manually force any mechanical movement.
Avoid to force, by any mechanical way, the movement of the wheels or any other part. Avoid to push the robot in a way that forces the wheels.

If you have any question or problem concerning the robot, please contact your Khepera dealer.

1.3 Recycling

Think about the end of life of your robot! Parts of the robot can be recycled and it is important to do so. It is for instance important to keep Ni-Cd batteries out of the solid waste stream. When you throw away a Ni-Cd battery, it eventually ends up in a landfill or municipal incinerator. These batteries, which contain heavy metals, can contribute to the toxicity levels of landfills or incinerator ash. By recycling the Ni-Cd batteries through recycling programs, you can help to create a cleaner and safer environment for generations to come. For those reasons please take care to the recycling of your robot at the end of its life cycle, for instance sending back the robot to the manufacturer or to your local dealer.

Thanks for your contribution to a cleaner environment!

2 UNPACKING AND INSPECTION

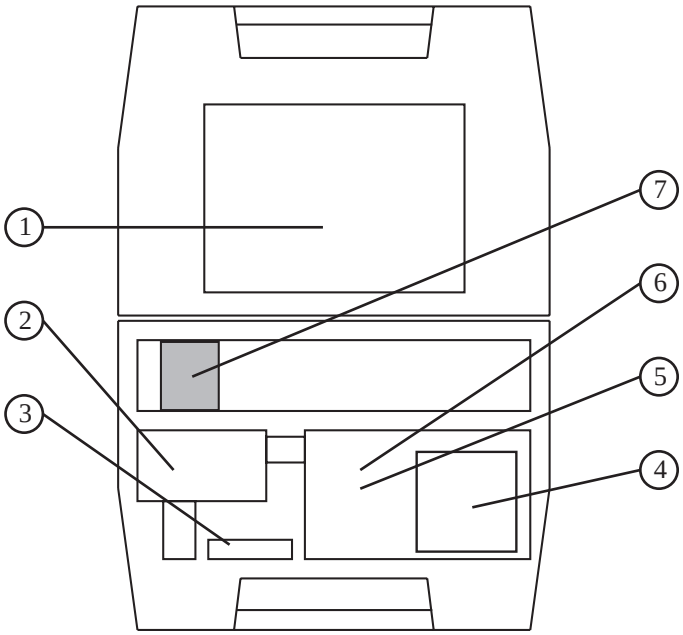


Figure 1: Position of the different parts in the bag.

Open the bag and check each item in the box against figure 1:

1. The documentation you are reading now
2. Power supply
3. Interface and charger module
4. Three disks with the software modules (VIs) for LabVIEW®
   - on SUN®
   - on Macintosh®
   - on PC
5. Cables:
   - Serial S
   - Battery
6. Spare parts: tires and jumpers
7. Your Khepera robot in the basic version

3 THE ROBOT AND ITS ACCESSORIES

3.1 The Khepera miniature robot

3.1.1 Overview

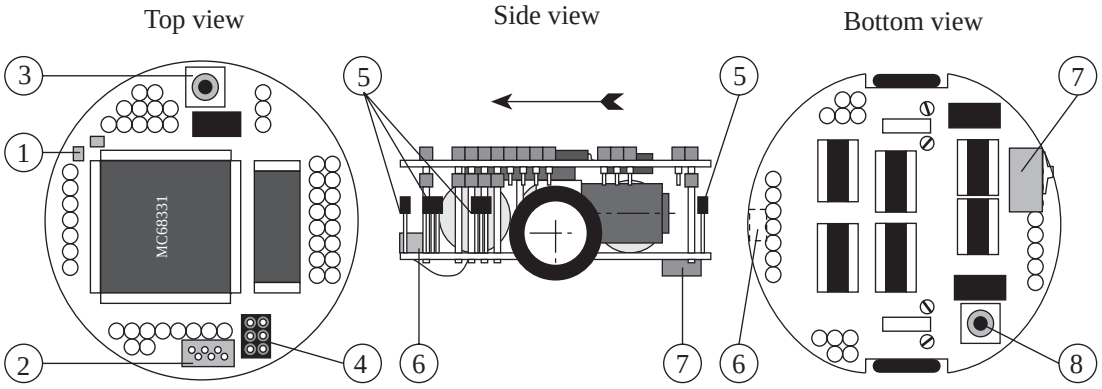


Figure 2: Position of some parts of the robot.

Make an external inspection of the robot. Note the location of the following parts:

1. LEDs
2. Serial line (S) connector.
3. Reset button.
4. Jumpers for the running mode selection.
5. Infra-Red proximity sensors.
6. Battery recharge connector.
7. ON - OFF battery switch.
8. Second reset button (same function as 3).

3.1.2 ON - OFF battery switch

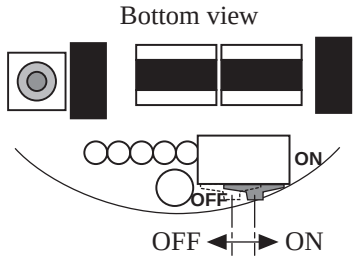


Figure 3: Position of the Battery power supply ON - OFF switch.

It allows the user to switch the battery of the robot ON or OFF. When ON, the robot is powered by the Ni-Cd internal batteries. In this case the robot cannot be powered

by an external supply. When OFF, the batteries are disconnected and the robot can be powered by the S serial line connector.

3.1.3 Jumpers, reset button and settings

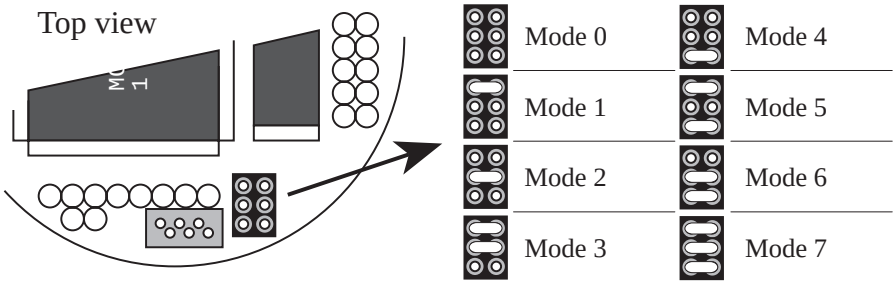


Figure 4: Position and setting of the jumpers

The ROM installed on your robot has an important library of software modules for the real time control of the Khepera robot. Part of these modules (building the BIOS) ensure the basic functionnalities of the Khepera robot, like motor control, sensors scanning etc. Another part of these modules ensure the interface with the user through the serial line. Depending on your use of the robot (remote control, downloading, test, demo etc.) you can select a specific module by setting the correspondent running mode. The 3 jumpers (see “Overview” on page 4) allow the selection of the most important running modes in several configurations. You have the choice between the following jumper configurations (see figure 4 for the corresponding jumper positions):

0. Demonstration mode: Braitenberg vehicle algorithm (number 3 according to the “Vehicle” book [Braitenberg84]) for obstacle avoidance.
1. Mode for the control of the robot by the serial communication protocol (see “The serial communication protocol” on page 18) using a serial link with a communication speed of 9600 Baud.
2. Same as mode 1 but with the communication speed of 19200 Baud.
3. Same as mode 1 but with the communication speed of 38400 Baud.
4. User application mode: start an application stored into the EPROM you can add to the standard modules (if any).
5. Downloading mode: in this mode the robot waits for a program to be transferred and executes it when downloaded (S format, 9600 Baud).
6. Same as mode 5 but with the serial link at 38400 Baud.
7. Test of the functionality of the robot. The result of successive tests is given on the serial link (9600 Baud).

The serial link set-up is always 8 bit, 1 start bit, 2 stop bit, no parity. Only the baud rate changes. The set-up of the jumpers can be changed at any time. **If the robot is running it is necessary to reset it to make the set-up effective.**

The reset button can be used at any time to reset the robot.

3.1.4 The S serial line

The S serial line is an asynchronous serial line with TTL levels (0-5V). An interface is necessary to connect this line to a standard RS232 port. This interface is included in the interface/charger module present in the package (see “Interface and charger module” on page 13). The S serial line can power the robot. **The length of the S serial cable should be limited to two meters for proper operation.**

3.1.5 Motors and motor control

Every wheel is moved by a DC motor coupled with the wheel through a 25:1 reduction gear. An incremental encoder is placed on the motor axis and gives 24 pulses per revolution of the motor. This allows a resolution of 600 pulses per revolution of the wheel that corresponds to 12 pulses per millimetre of path of the robot.

The Khepera main processor has the direct control on the motor power supply and can read the pulses of the incremental encoder. An interrupt routine detects every pulse of the incremental encoder and updates a wheel position counter.

The motor power supply can be adjusted by the main processor by switching it ON and OFF at a given frequency and during a given time. The basic switching frequency is constant and sufficiently high not to let the motor react to the single switching. By this way, the motor react to the time average of the power supply, which can be modified by changing the period the motor is switched ON. This means that only the ratio between ON and OFF periods is modified, as illustrated in figure 5. This power control method is called “pulse width modulation” (PWM). The PWM value is defined as the time the motor is switched ON.

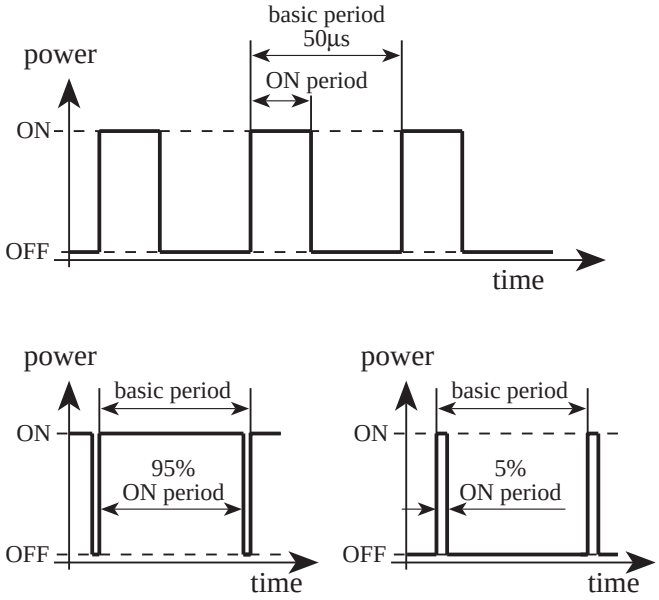


Figure 5: The “pulse width modulation” (PWM) power supply mode is based on a ratio between the ON time and the total time. The basic switching frequency is constant.

The PWM values can be set directly, or can be managed by a local motor controller. The motor controller can perform the control of the speed or position of the motor, setting the correct PWM value according to the real speed or position read on the incremental encoders.

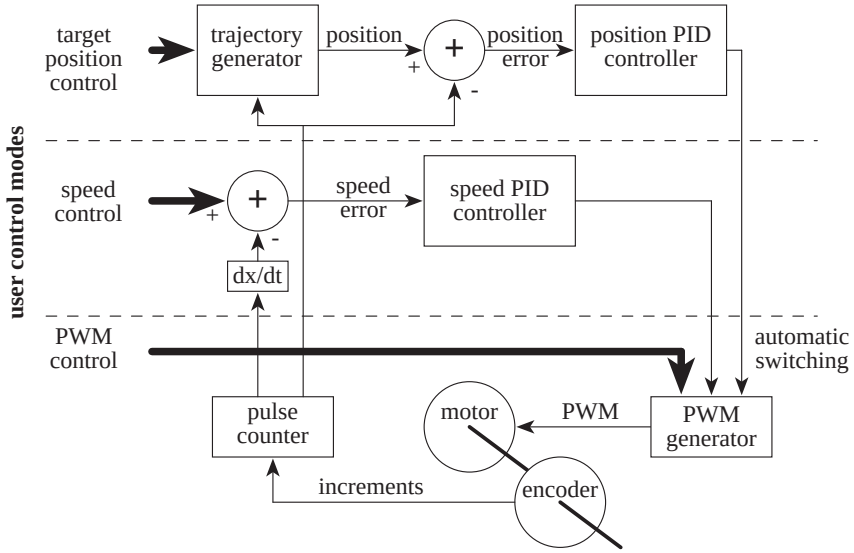


Figure 6: Structure of the motor controllers and levels of user access.

Both DC motors can be controlled by a PID controller executed in an interrupt routine of the main processor. Every term of this controller (Proportional, Integral, Derivative) is associated to a constant, setting the weight of the corresponding term: K_p for the proportional, K_i for the integral, K_d for the derivative.

The motor controller can be used in two control modes: The speed and the position modes. The active control mode is set according to the kind of command received. If the controller receives a speed control command, it switches to the speed mode. If the controller receives a position control command, the control mode is automatically switched to the position mode. Different control parameters (K_p, K_i and K_d) can be set for each of the two control modes.

Used in speed mode, the controller has as input a speed value of the wheels, and controls the motor to keep this wheel speed. The speed modification is made as quick as possible, in an abrupt way. No limitation in acceleration is considered in this mode.

Used in position mode, the controller has as input a target position of the wheel, an acceleration and a maximal speed. Using this values, the controller accelerates the wheel until the maximal speed is reached, and decelerates in order to reach the target position. This movement follows a trapezoidal speed profile, as described in figure 7.

The input values and the control mode of this controller can be changed at every moment. The controller will update and execute the new profile in the position mode, or control the wheel speed following the new value in the speed mode. A status of the controller indicates the active control mode, the phase of the speed profile (on target or in movement) and the position error of the controller.

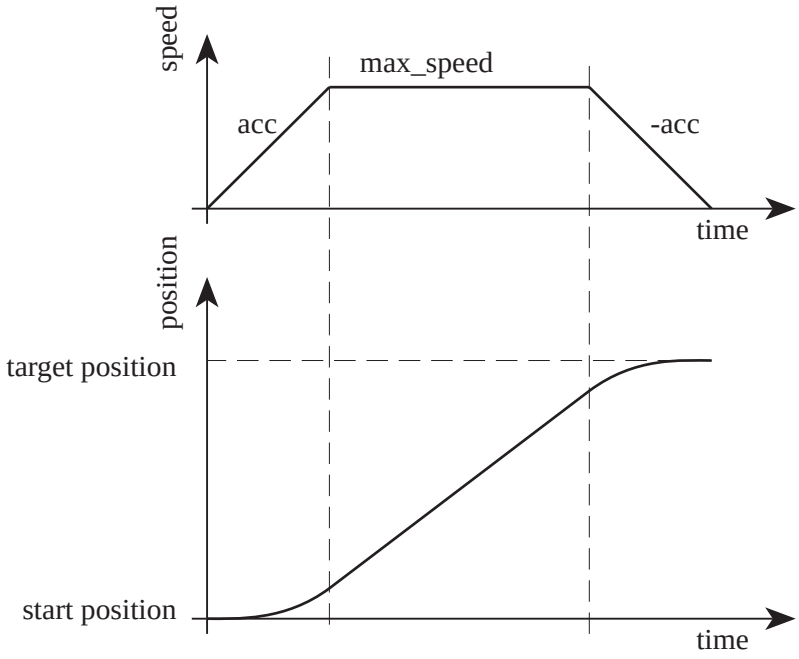


Figure 7: Speed profile used to reach a target position with a fixed acceleration (acc) and a maximal speed (max speed).

3.1.6 Infra-red proximity sensors

Eight sensors are placed around the robot and are positioned and numbered as shown in figure 8.

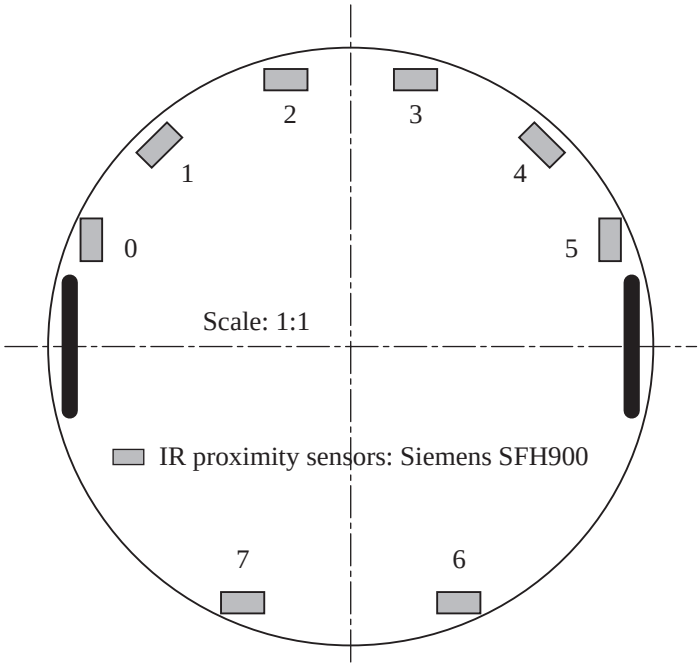


Figure 8: Position of the 8 IR sensors.

These sensors embed an infra-red light emitter and a receiver. For more information about these particular devices, please refer to the documentation of the sensor manufacturer. The exact part name is SFH900-2 and the manufacturer is SIEMENS.

This sensor device allows two measures:

- The normal ambient light. This measure is made using only the receiver part of the device, without emitting light with the emitter. A new measurement is made every 20 ms. During the 20 ms, the sensors are read in a sequential way every 2.5 ms. The value returned at a given time is the result of the last measurement made.
- The light reflected by obstacles. This measure is made emitting light using the emitter part of the device. The returned value is the difference between the measurement made emitting light and the light measured without light emission (ambient light). A new measurement is made every 20 ms. During the 20 ms, the sensors are read in a sequential way every 2.5 ms. The value returned at a given time is the result of the last measurement made.

The output of each measurement is an analogue value converted by a 10 bit A/D converter. The following two sections (3.1.6.1 and 3.1.6.2) illustrate the meaning of this 10 bit values.

3.1.6.1 Ambient light measurements

The measurement of the ambient light versus the distance of a light source has the following shape:

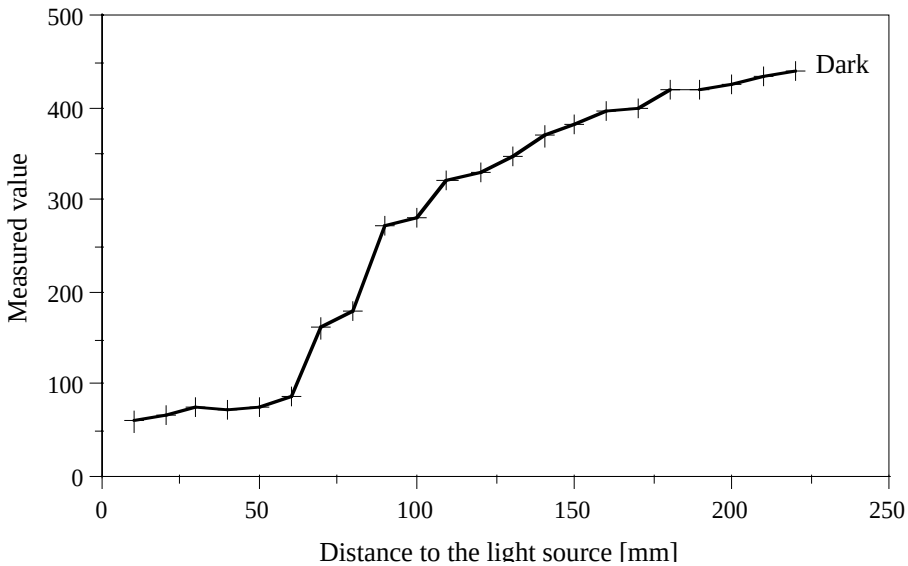


Figure 9: Typical measurement of the ambient light versus the distance of a light source of 1 Watt.

As it can be seen, the measured value decreases when the intensity of the light increases. The standard value in the dark is around 450.

The measurement of the ambient light versus the angle between the forward direction of the robot and the direction of the light has the shape illustrated in figure 10.

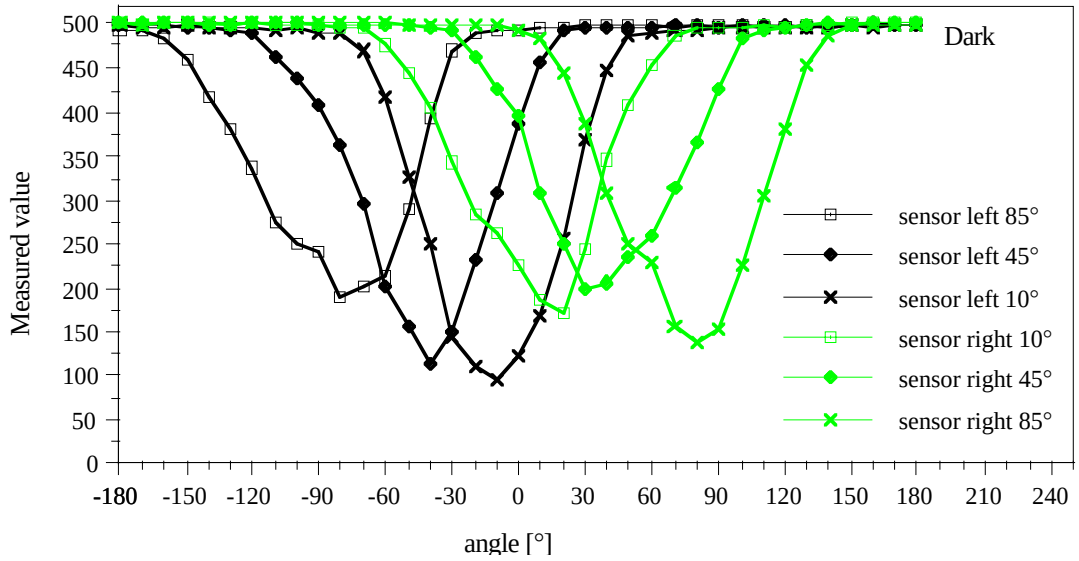


Figure 10: Typical measurement of the ambient light with a light source moving around the robot. The angle on the X axis is measured between the forward direction of the robot and the direction of the light.

All these measurements depend very strongly from various factors like the distance of the light source, the colour, the intensity, the vertical position, etc. These two figures show only the global shape of the sensor's response.

3.1.6.2 Reflected light measurements

The measurement of the proximity of obstacles by reflected light depends on two major factors: the reflexivity of the obstacle (colour, kind of surface...) and the ambient light. Figure 9 shows some measurements giving an idea of the response of the sensor.

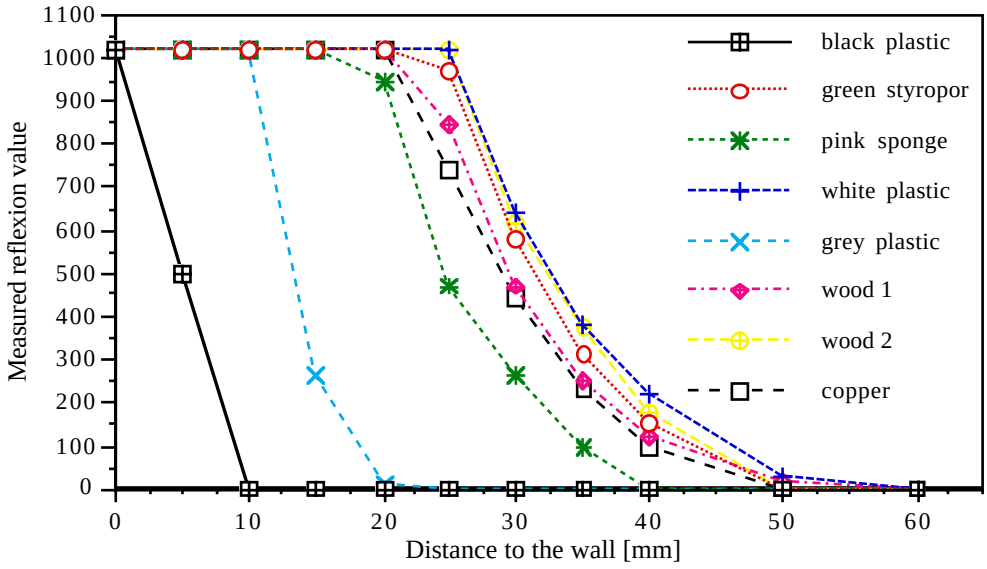


Figure 11: Measurements of the light reflected by various kinds of objects versus the distance to the object.

The directionality of the sensor measurement is illustrated in Figure 10: these sensors have a very large field of view.

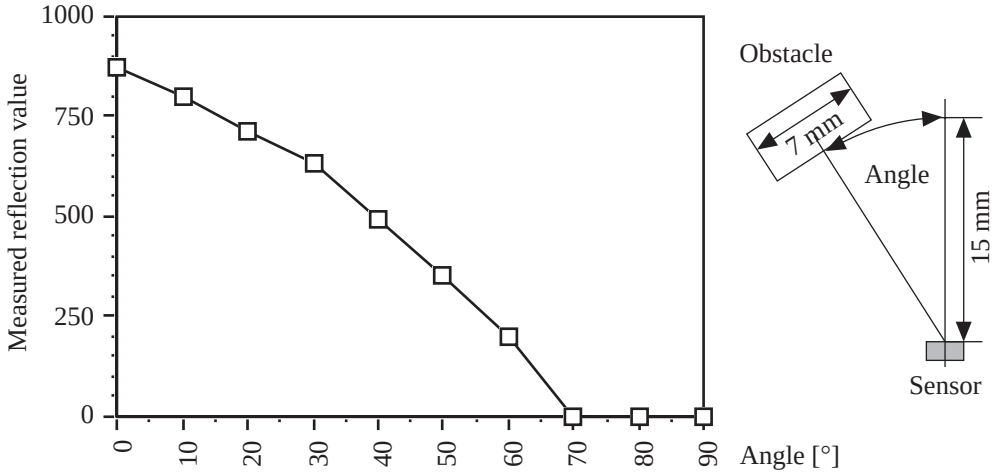


Figure 12: Typical response of a proximity sensor for an obstacle (7 mm in width) at a distance of 15 mm. The measurement is given versus the angle between the forward orientation of the robot and the orientation of the obstacle.

The characteristics of the different physical sensors can change in a large range. Figure 13 shows the measurements made on six sensors of the same robot placed in identical conditions. Small differences of conditions (vertical orientation of the sensor, ambient light conditions, colour of the floor) can bring additional differences.

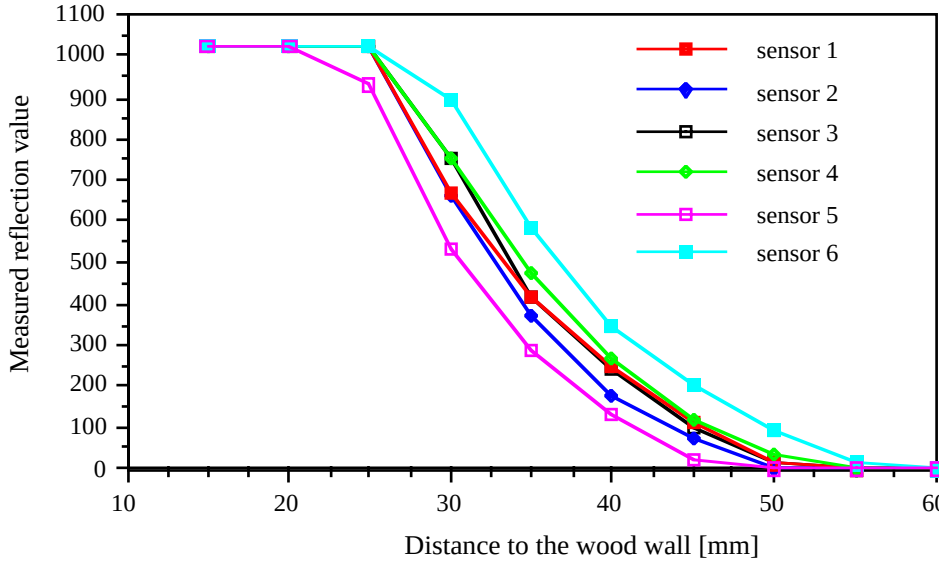


Figure 13: Typical measurements of the light reflected by a wall versus the distance to the wall for several sensors of the same kind and in the same conditions.

3.1.7 Batteries

The robot is equipped with 4 rechargeable Nickel-Cadmium batteries with a capacity of 180 mAh (older versions can have 110 mAh). This capacity allows the robot an autonomy of about 45 minutes in the basic configuration. The battery can be charged with the interface/charger module (see “Interface and charger module” on page 13)

There is no specific management of the battery charge level. During a life cycle of the robot, the battery discharges as indicated in figure 14. When the battery voltage is under 4 V, the robot processor stops working correctly and the robot has no more control. At this voltage there is still sufficient power to make the motors move, which means that the robot can move without control.

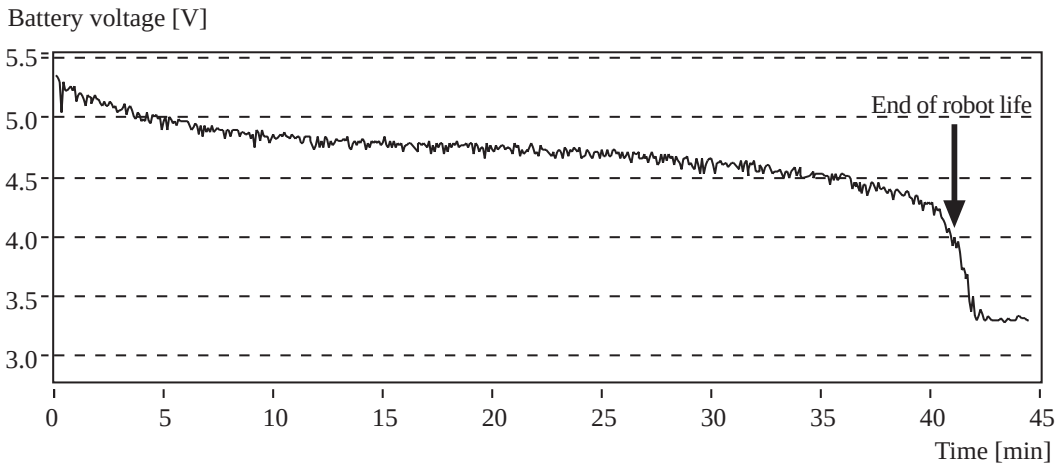


Figure 14: Battery discharge cycle

3.2 Cables and accessories

The S cable (2 m long, with a 6 pins connector) allows the connection between the robot and the interface/charger module to support the communication to the host computer (see “Interface and charger module” on page 13).

The recharging cable (0.5 m long, with a 4 pins connector) allows the connection between the robot and the interface/charger module to recharge the robot.

Some exchange parts are also included in the package: 4 new tires, 10 jumpers.

3.3 Power supply

A transformer provides the power supply to the interface module and, if the battery switch of the robot is OFF, to the robot itself. The connection between the power supply and the interface/charger module is made by the power supply jack (see “Interface and charger module” on page 13).

SAFETY PRECAUTION: The power supply must be connected to the wall socket only when all other connections are already made.

3.4 Interface and charger module

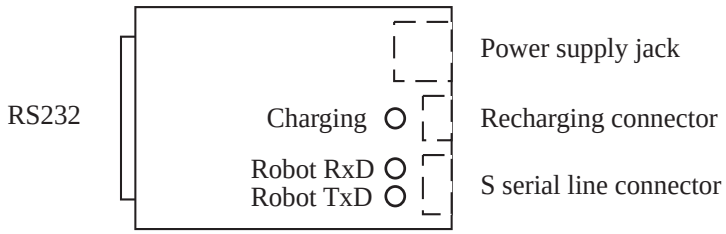


Figure 15: The interface - charger module

This module supports the interface between the robot and the host computer, the power supply of the robot from the S serial line cable and the charge of the battery. To work, the module needs to be connected to the power supply.

The following features are present in this module:

- **The battery charger**: a four pins connector allows the connection with the Khepera robot to charge its battery. **When charging, the robot must be disconnected to all other systems and switched OFF. Avoid to recharge full batteries, this can cause damages!** It would be optimal to discharge the robot completely (leave the robot swathed on) before recharging it! During the charging period, a yellow led indicates the activity. The charging time for an empty battery is about 45 minutes. See “Charging configuration” on page 14 for more details.
- **The S - RS232 interface**: this interface allows the connection between the robot (S serial line) and a host computer (over a RS232 port). Two connectors are available: a standard female DB25 (the interface module is a DCE) for the RS232 link toward the host computer and a S six pins connector for the link toward the robot.

The link towards the robot also powers the robot if the battery switch is OFF. See “Configuration for robot-computer communication” on page 15 for more detail on this working configuration.

3.5 Software support floppy disks

Three 3,5″ floppy disks contain all the modules for interfacing the Khepera robot with LabVIEW® on PC, Macintosh® and SUN® (see “Using LabVIEW®” on page 22). The software LabVIEW® is a product of National Instruments and is not included in the package.

4 UNPACKING TEST

After unpacking it is important to test the functionality of the robot. A test that uses most of the possible functionnalities is available with the running mode 0: a Braitenberg vehicle (see “Jumpers, reset button and settings” on page 5). To obtain this running mode operate as following:

- Put the robot on a flat surface without danger for the robot. Water, edge of the table or metallic objects have to be considered as dangerous for the robot. Be aware that the robot will move rather quickly and cover long distances in a short delay. The robot is normally charged when delivered.
- Verify that the three jumpers are not connected (setting for the running mode 0, see “Jumpers, reset button and settings” on page 5).

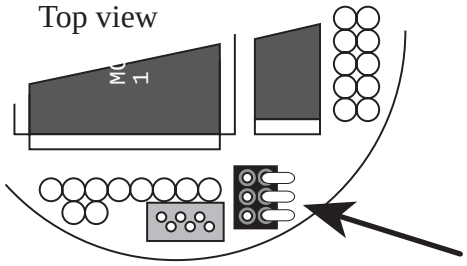


Figure 16: Settings of the jumpers for the Braitenberg vehicle test.

- Switch the robot battery switch ON and put the robot on the surface.

The robot must start to go forward while avoiding obstacles. The obstacles must be bright to better reflect the light of the proximity sensors. If the robot does not operate properly, check the three points mentioned above, recharge the robot and retry. If the robot does not correctly avoid the obstacles, please contact your Khepera dealer.

5 CONNECTIONS

There are two standard configurations: a first one used to charge the robot and a second one that allows the communication between the robot and the host computer.

5.1 Charging configuration

Warning: It is necessary to discharge the batteries before recharging. Avoid to start a recharging process on charged batteries. This can cause damages.

To charge the battery of the robot, the following connections have to be made:

- Between the robot and the interface/charger module with the charging cable (4 pins).
- Warning: the robot battery switch must be on the OFF position.

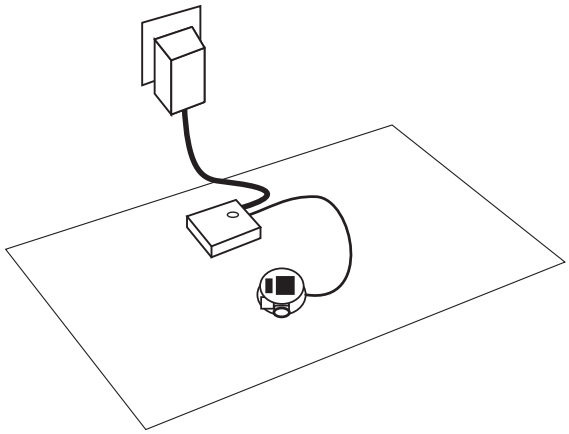


Figure 17: Connections to recharge the robot's batteries.

- Between the interface/charger module and the power supply with the jack.
- Plug the power supply to the wall socket only when these connections are established.

The charger will perform some measurements and then start to charge. These operations can take 10 minutes if the batteries are hot (after long use) or too discharged. When charging, the "charging" indication led is ON. If the led does not switch ON after 10 minutes, unplug and re-plug the power supply.

The led is switched OFF at the end of the charging process. The charging time for an empty battery is about 40 minutes. At this moment you can unplug the power supply and remove the charger cable. When charging, the battery can be hot (50°C). This is normal.

5.2 Configuration for robot-computer communication

This configuration allows the communication between the robot and a host computer through a serial link. On the host computer side the link is made by a RS232 line. The interface module converts the RS232 line into the S serial line available on the robot.

The following connections must be made:

- Between the robot and the interface/charger module by the S serial cable. This cable also supports the power supply of the robot. This external power supply is available when the general battery switch is OFF. If the switch is ON, the robot uses its own batteries for power supply.
- Between the interface module and the host computer by a standard RS232 cable. This cable is not in the package because there are several standards at the level of the host connector. You can easily find this cable by your host computer dealer. If your host computer RS232 port has a DB25 male connector you can plug the interface module directly in it.

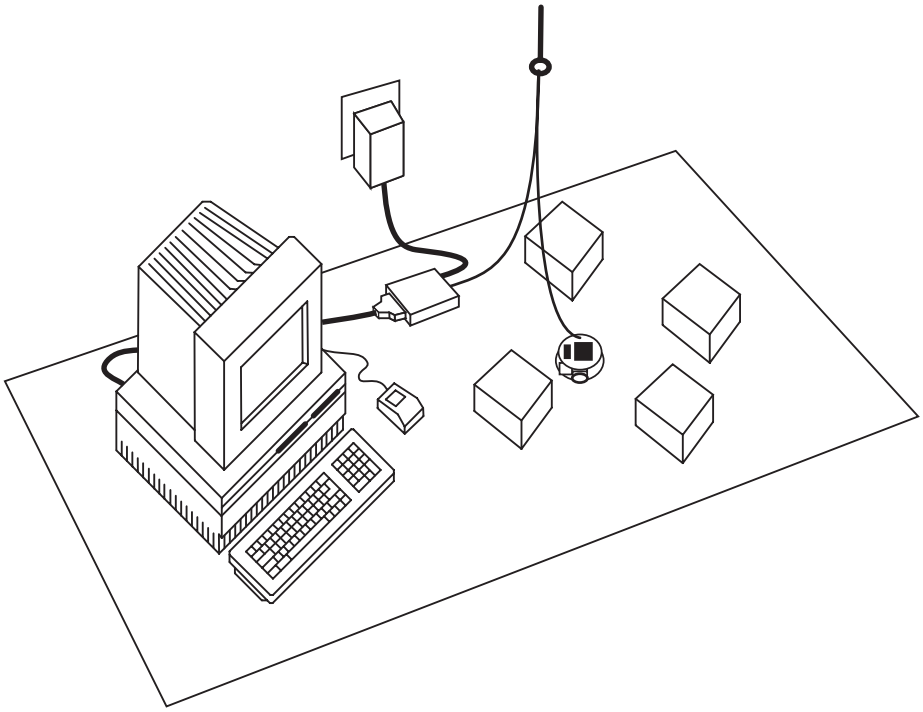


Figure 18: Configuration for the communication between the robot and the host computer.

- Between the interface/charger module and the power supply with the cable fixed to the power supply.
- Set the jumpers according to the desired running mode (see “Jumpers, reset button and settings” on page 5). Be careful that in running mode 0 the robot starts moving when powered.
- Plug the power supply to the wall socket.

To test the connection and the settings of the serial port do the following manipulations:

- Put the robot on a flat surface on which the robot can move around easily. The battery switch must be OFF.
- Insert all jumpers like in figure 19 (setting for the running mode 0, see “Jumpers, reset button and settings” on page 5).

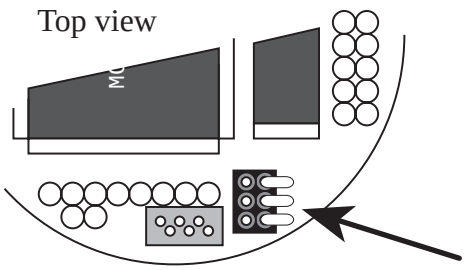


Figure 19: Settings of the jumpers for the demo mode.

- Run a terminal emulator on your host computer (for instance Hyper Terminal on PCs, Microphone on MACs, tip on UNIX or minicom on linux, see “RS232 configuration” on page 45) connected to the serial port on which you have connected the robot. **Configure your terminal as following: 9600 Baud, 8 bit, 1 start bit, 2 stop bit, no parity.**
- Plug the mural power supply to the main, or, if it is already connected, **reset the robot** pressing on the reset button.

Your terminal should display:

```
ROM of minirobot KHEPERA,...
```

The transmit data (Robot TxD on the interface module) green lamp should blink after reset. If the robot does not respond as indicated (the green led does not blink after reset), check the points mentioned above and retry. If the green led blinks but your computer does not show any message, check the configuration of your serial port and terminal emulator, as well as the connection between interface/charger module and host computer.

6 THE SERIAL COMMUNICATION PROTOCOL

The serial communication protocol allows the complete control of the functionnalities of the robot through an RS232 serial line. It correspond to running modes 1 to 3 (see “Jumpers, reset button and settings” on page 5). The connection configuration necessary to use these functionnalities is presented in the section 5.2 of this manual. The configuration (baudrate as well as data, start, stop and parity bits) of the serial line of your host computer must correspond to the one set on the robot with the jumpers (running modes 1 to 3, always 8 bit, 1 start bit, 2 stop bit, no parity).

The communication between the host computer and the Khepera robot is made sending and receiving ASCII messages. Every interaction is composed by:

- A command, sent by the host computer to the Khepera robot and followed by a carriage return or a line feed.
- When needed, a response, sent by the Khepera to the host computer.

In all communications the host computer plays the role of master and the Khepera the role of slave. All communications are initiated by the master.

The communication is based on two types of interactions: one type of interaction for the set-up of the robot (for instance to set the running modes, the transmission baudrate...), and one type of interaction for the control of the functionality of the robot (for instance to set the speed of the motors, to get the values of the sensors...). The interactions allowing the set-up of the robot are based on commands called *tools*. The interactions for the control of the robot functionality uses *protocol* commands and responses.

6.1 The tools

Here is the description of some basic tools:

“run”	Starts a function stored in the ROM. It has to be followed by the function name. The functions available in the ROM can be listed with the *list* tool and have an identification string beginning with “FU”. Some of the functions correspond to the running modes presented in the section 3.1.3. Using the *run* tool it is possible, for instance, to start the demo mode (Braitenberg vehicle, corresponding to the running mode 0, as described in section 3.1.3) typing “run demo” and return.
“serial”	Sets the serial channel to a baud-rate, given as parameter. For instance, typing “serial 19200” and return you set the baudrate to 19200 Baud.
“help”	Shows the help message of the modules available in the ROM. A parameter can be used to indicate a need of help on a particular item. “help list” gives an help message for the *list* tool. “help demo” gives an help message on the *demo* function mentioned above.
“list”	Gives the list of all the tools, functions, protocol commands

	and other modules available in ROM. For every item listed you get an ID, a name, a description and a version. The ID is composed by four letters. The first two letters define the family of the module: IDs starting by “TA” define tasks running on Khepera, “FU” functions that can be executed with the *run* command, “PR” protocol commands, “TO” tools like this one and “BI” BIOS components.
“k-team”	Gives a short description of the K-Team active members.
“net”	Gives an information about the intelligent extension turrets installed on the robot. For each item listed you get a name, an ID (to be used to address the turret, see also the command ‘T’ in appendix A), a description and a revision.
“memory”	Gives an information about the memory used by the system.
“restart”	Resets the robot. This action is equivalent to a hardware reset.
“process”	Gives the list of all processes running on Khepera in parallel to the serial communication protocol management.
“sfill”	Starts a Motorola S format downloader. This downloader does not start the execution of the code at the end of the download. To download and execute a code, please use the sloader function, started by the command “run sloader”.

6.2 The control protocol

To control the functionnalities of the Khepera robot (motors, sensors etc.), a set of command are implemented in the control protocol. Also in this case, the communication with the Khepera robot is made sending and receiving ASCII messages. Every interaction between host computer and Khepera is composed by:

- A command, beginning with one or two ASCII capital letters and followed, if necessary, by numerical or literal parameters separated by a comma and terminated by a carriage return or a line feed, sent by the host computer to the Khepera robot.
- A response, beginning with the same one or two ASCII letters of the command but in lower-case and followed, if necessary, by numerical or literal parameters separated by a comma and terminated by a carriage return and a line feed, sent by the Khepera to the host computer.

In all communications the host computer plays the role of master and the Khepera the role of slave. All communications are initiated by the master.

The protocol commands are 18 and a complete description is given in Appendix A.

6.3 Testing a simple interaction

To better understand both tools and protocol commands, we propose to do a very simple test as following:

- Set the jumpers to select the running mode 1 (see “Jumpers, reset button and settings” on page 5)

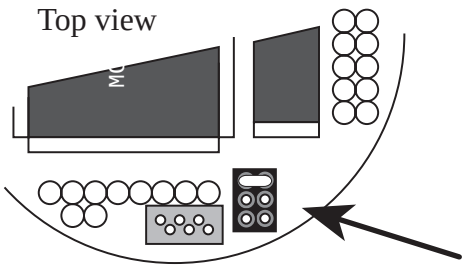


Figure 20: Settings of the jumpers for the 9600 baud communication mode.

- Set the connection configuration presented in section 5.2.
- Start on your host computer a terminal emulator (for instance Hyper Terminal on PCs, Microphone on MACs, tip on UNIX or minicom on linux, see “RS232 configuration” on page 45) with the serial line set to 9600 Baud, 8 bit data, 1 start bit, 2 stop bit, no parity.

We start testing some protocol commands:

- Type the capital letter **B** followed by a carriage return or a line feed.
- The robot must respond with **b** followed by an indication of the version of software running on the robot and terminated by a line feed.
- Type the capital letter **N** followed by a carriage return or a line feed.
- The robot must respond with **n** followed by 8 numbers separated by a comma and terminated by a line feed. These numbers are the values of the robot proximity sensors presented in section 3.1.6.
- Retry the same command (N) putting some obstacles in front of the robot. The response must change.
- Type the protocol command **D,5,-5** followed by a carriage return or a line feed.
- The robot must start turning on place and respond with **d** and a line feed.
- To stop the robot type the protocol command **D,0,0** followed by a carriage return or a line feed.
- Type the protocol command **H** followed by a carriage return or a line feed.
- The robot must respond with **h** followed by 2 numbers separated by a comma and terminated by a line feed. These numbers are the values of the position counters of each wheel.
- Type the protocol command **G,0,0** followed by a carriage return or a line feed.

- This command set the position counters to the 2 values given as parameters. The answer is composed by a **g** and a line feed.
- Retry the protocol command **H** to verify that the G command has been executed.
- Type the protocol command **C,1000,1000** followed by a carriage return or a line feed.
- The robot respond with **c** and goes forward 80 mm.
- Retry the protocol command **H** to verify the final position.
- Try other commands following the description given in Appendix A.

We continue testing some tools:

- Type the command **help** followed by a carriage return or a line feed.
- The robot must respond with the list of all tools available.
- Type the command **help serial** followed by a carriage return or a line feed.
- The robot must respond with the description of the “serial” tool.
- Type the command **help D** followed by a carriage return or a line feed.
- The robot must respond with the description of the “D” protocol command.
- Type the command **list** followed by a carriage return or a line feed.
- The robot must respond with the list of all software modules present in the ROM. In addition to the “tools” (characterised by a “TOXX” ID) and the “protocol” commands (characterised by a “PRXX” ID) you can find on the list “functions” (characterised by a “FUXX” ID) and BIOS modules (characterised by a “BIXX” ID). On every module you can have an help message.

7 USING LABVIEW®

This chapter is to familiarise you with the LabVIEW® environment in the context of Khepera use. LabVIEW® is a product of National Instruments (http://www.natinst.com). To this end, the examples are presented in an increasing order of complexity. Our advice is to follow the chronological order of presentation. Please refer to the LabVIEW manuals for more information about this software.

The following examples and the files distributed with this product are based on LabVIEW® version 5.

LabVIEW® runs on your PC, Macintosh® or SUN® workstations, and can control the functionality of the Khepera robot using the serial communication protocol described in section 6.2.

7.1 Hardware configuration

Set your environment as illustrated in section 5.2. The jumpers must be set as showed in figure 21, to obtain the running mode 2 (for more details on running modes please refer to section 3.1.3).

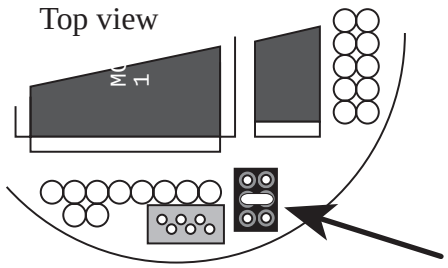


Figure 21: Settings of the jumpers for using LabVIEW ® and a serial connection at 19200 baud.

7.2 Set up of the serial link

To enable the exchange of information between your computer and the robot, you have to configure the serial link of your host computer, according to the osetting chosen on the Khepera robot.

Be sure that the connection cable is connected at both ends (Khepera and interface), that the robot is powered (power adaptor), then start LabVIEW® and open the Set-up virtual instrument (called “VI”) present in your floppy disk.

The panel illustrated in figure 22 should appear.

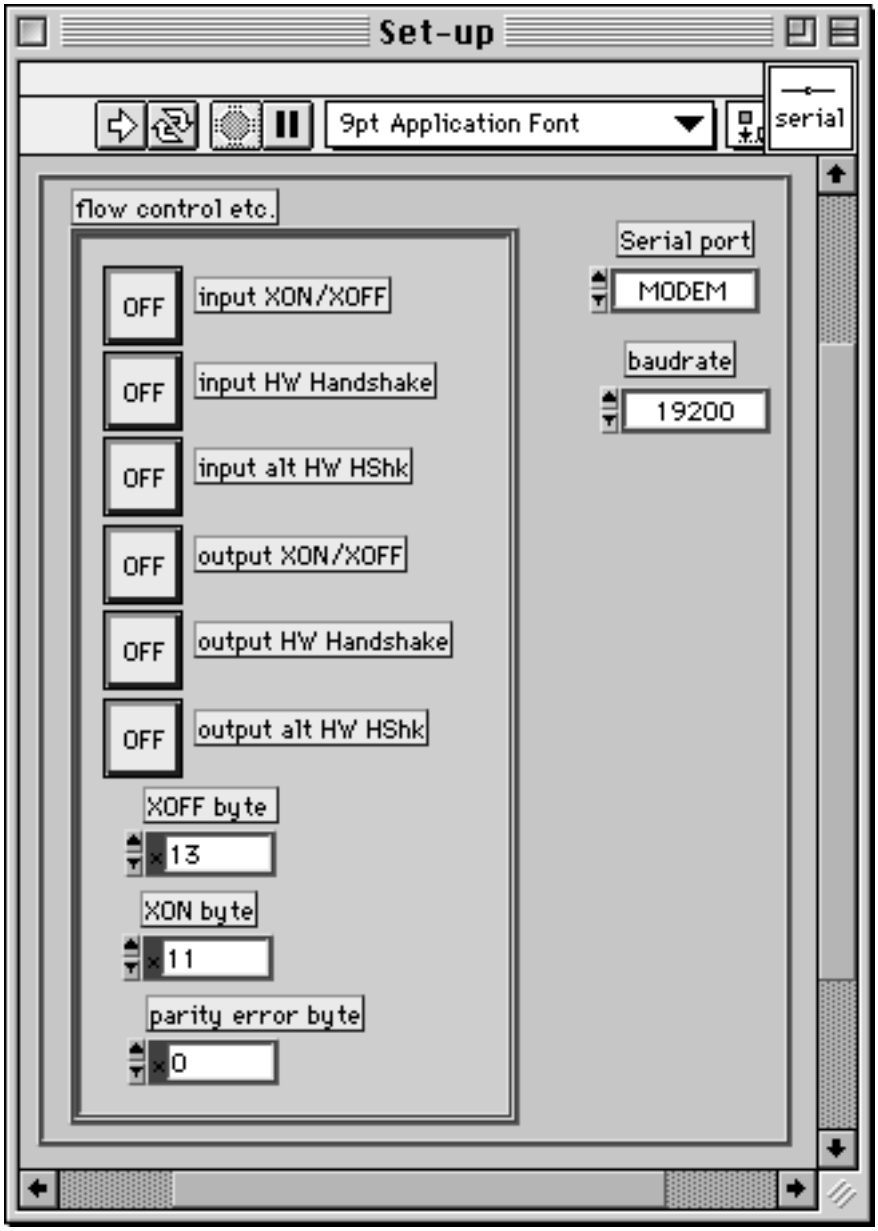


Figure 22: Set up panel for serial link initialisation.

Now, select the serial port on which the robot is connected. This selection depend on which port you use and the type of computer you have. This choice must be made for every module that you will use.

Then click once on the run arrow  at the top of the window.

A stop icon  appears for a few seconds, after what the front panel returns to its initial state.

That's all! The serial link with Khepera is set to 19200 baud. It will remain so until you quit LabVIEW®.

7.3 Motors

We will now control the displacement of the robot. Be sure that the serial link has been correctly installed, then open the Motors VI present on the floppy disk. Now your screen displays the following panel:

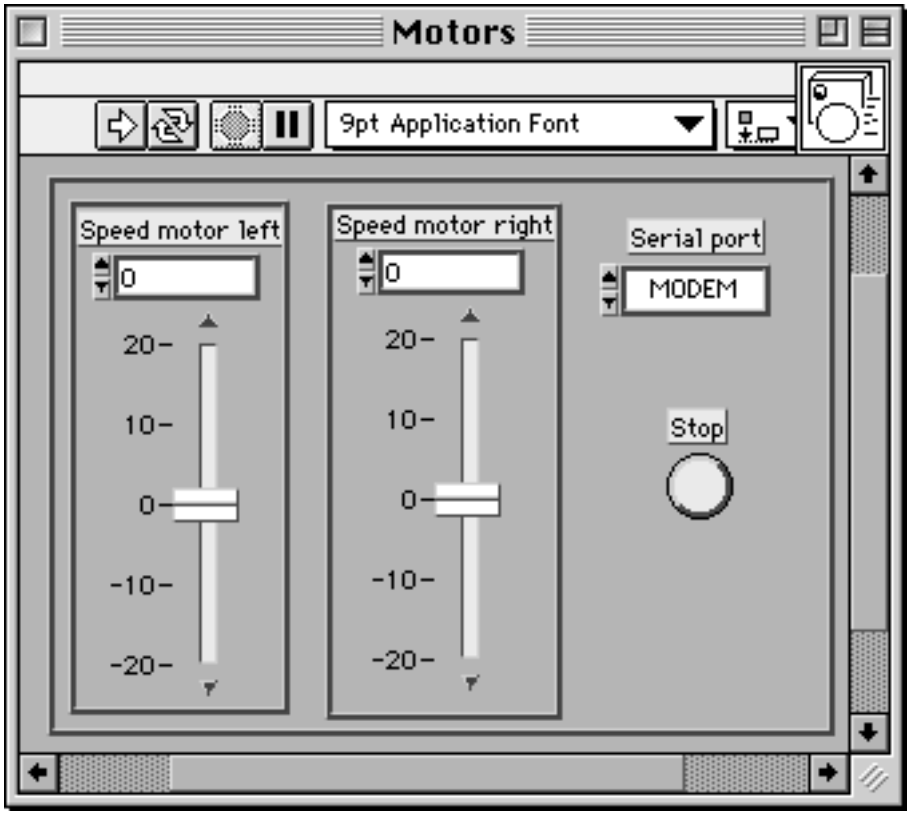


Figure 23: Motors panel: 2 sliders controlling the speed of each wheel.

Before to move the robot, you must learn how to stop it. Different means are available. Starting from the most efficient:

- Press the reset button on the robot once.
- Put the value 0 for each speed using the sliders. Don't forget that these new values will only be taken into account at the next execution (i.e., click on the arrow).
- Click on the button labelled Stop. You have also to click on the arrow again so that the robot takes your last decision into account. Before trying to give another values to the motors, click again on the button (to de-select this option). This last option is the best way to stop the robot.

You control directly each one of the motors by simply putting the desired speed values in the corresponding slider. This can be made moving the slider or entering the desired speed in the digital display placed between the sliders and their names.

Possible values are constrained (only on the sliders) between -20 and +20 so to take care of the mechanics. To transmit your order to the robot, just click once on the arrow. You can change the values and click on the arrow again to validate your choice. You see that the robot continues moving at the same speed until new values are send.

If you are getting bored with clicking on the arrow, try one click on the double arrow  . Click on the stop icon to stop the execution.

The robot has two incremental encoders on the wheels. Using these sensors it is possible to measure the displacement and the speed of each wheel at every moment. The VI “Get_position” ask for the content of the increment counter, which represent the displacement of the wheel. The unit of displacement correspond to 0.08mm.

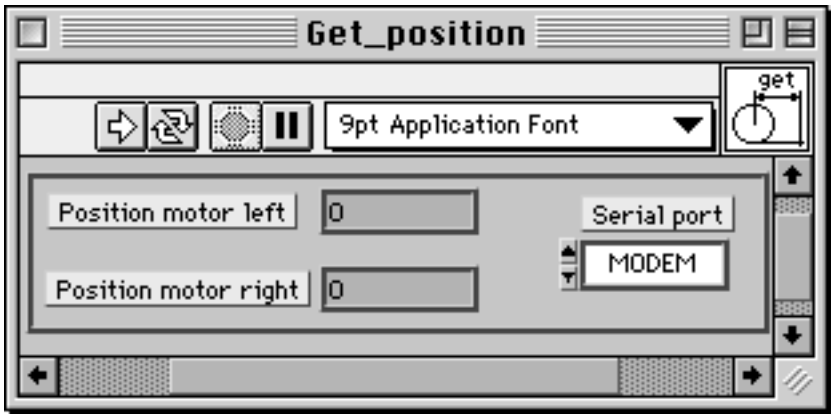


Figure 24: Get_position panel: 2 indicators show the position of each wheel.

To test the functionality of this module just click on the double arrow to start the recurrent running mode. At this moment the VI will show you the actual position of each wheel. To change the position of the wheel use the “Motors” VI as described above, and observe the result on the “Get_position” VI. To set the position counter to a given value you can use the “Set_position” VI.

The “Get_speed” VI display the speed of each wheel. This value is computed on the robot, based on the information of the position and the time.

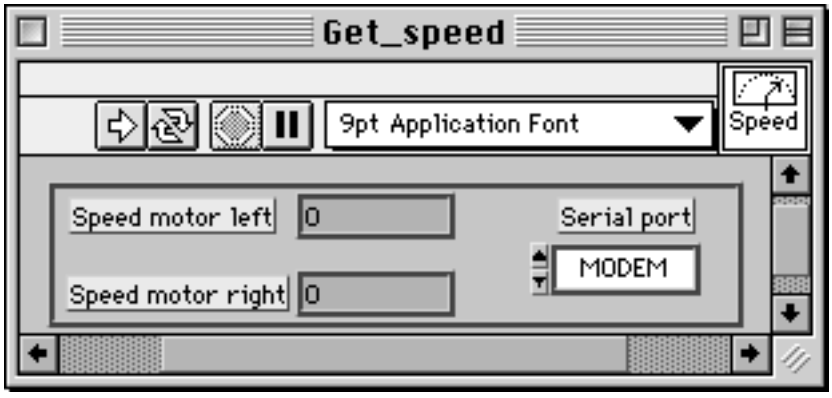


Figure 25: Get_speed panel: 2 indicators show the speed of each wheel.

To test the functionality of this module just click on the double arrow to start the recurrent running mode. At this moment the VI will show you the actual speed of each motor. To change the speed of the motors use the “Motors” VI as described above. You can also try to set the motor speed to 5 using the “Motors” VI, then slow down the wheels with your fingers and look to the result on the “Get_speed” VI.

It is also possible to give to the robot a position to reach, expressed using the position of the two wheels as described above. The position given as target will be reached using a trapezoidal speed profile (as described in section 3.1.5 of this manual). The target position can be given to the robot with the “Control_position” VI, illustrated in figure 26. Also here the commands can be made moving the slider or entering the desired position in the digital display placed between the sliders and their names.

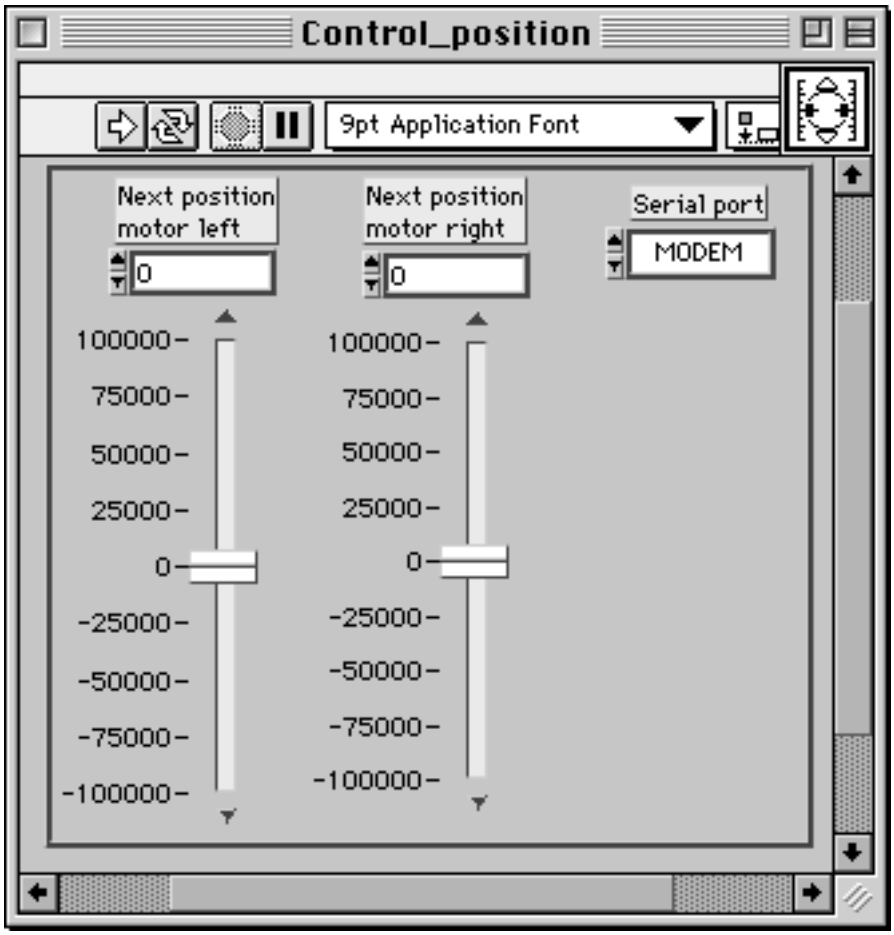


Figure 26: Control_position panel: 2 sliders controlling the position to reach for every wheel.

As described in section 3.1.5 of this manual, every displacement in position control mode will be made following a precise speed profile. The parameters of this speed profile can be configured using the VI “Conf_pos_param”, showed in figure 27. For each motor you can set acceleration and maximal speed. Deceleration will be set identical to acceleration. Figure 27 shows default values.

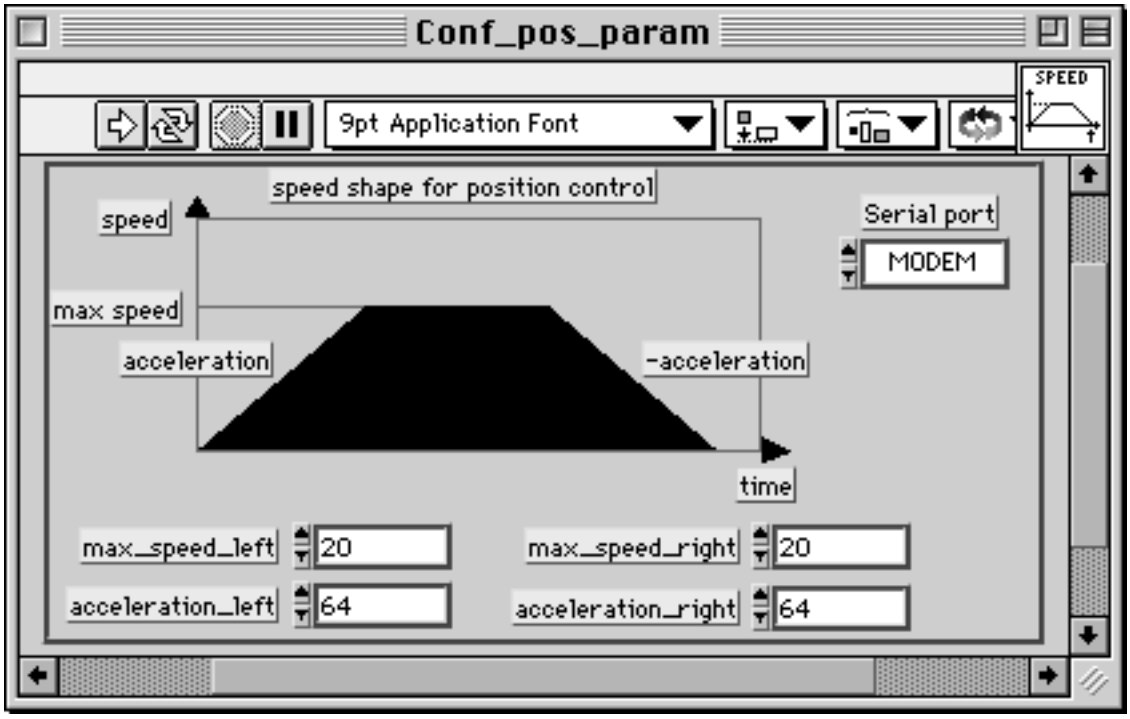


Figure 27: Conf_pos_param panel: all the parameters of the speed profile can be controlled.

To test the position control, please start setting to 0 the two position counters of the two wheels, using the “Set_position” VI or resetting the robot. At this moment you can try the “Control_position” VI setting a distance of 1000 on both sliders and running the VI once, clicking on the run arrow. The robot should move forward showing an acceleration, a fixed speed and then a deceleration. Test other movements keeping both left and right displacements identical to move on a line.

To start other kinds of trajectory, bring back the robot to the 0 position or use the “Set_position” VI to reset the counter. Then use the “Conf_pos_param” VI to set the left maximal speed to 10, the left acceleration to 32, and keeping the values of the right trajectory to the default value indicated in figure 27. Run the VI once to make these values effective. Then set, on the “Control_position” VI, the goal position of the left wheel to 1000 and the goal position of the right wheel to 2000. Run the VI once and observe the trajectory of the robot. The robot should make a circular trajectory.

7.4 Sensors

In its basic version, Khepera has eight infra-red sensors, as described in section 3.1.6. You will now easily understand their characteristics. Be sure of the set-up of the serial link, then open the Sensors VI that you have on the floppy disk. You should see on your screen the panel illustrated in figure 28.

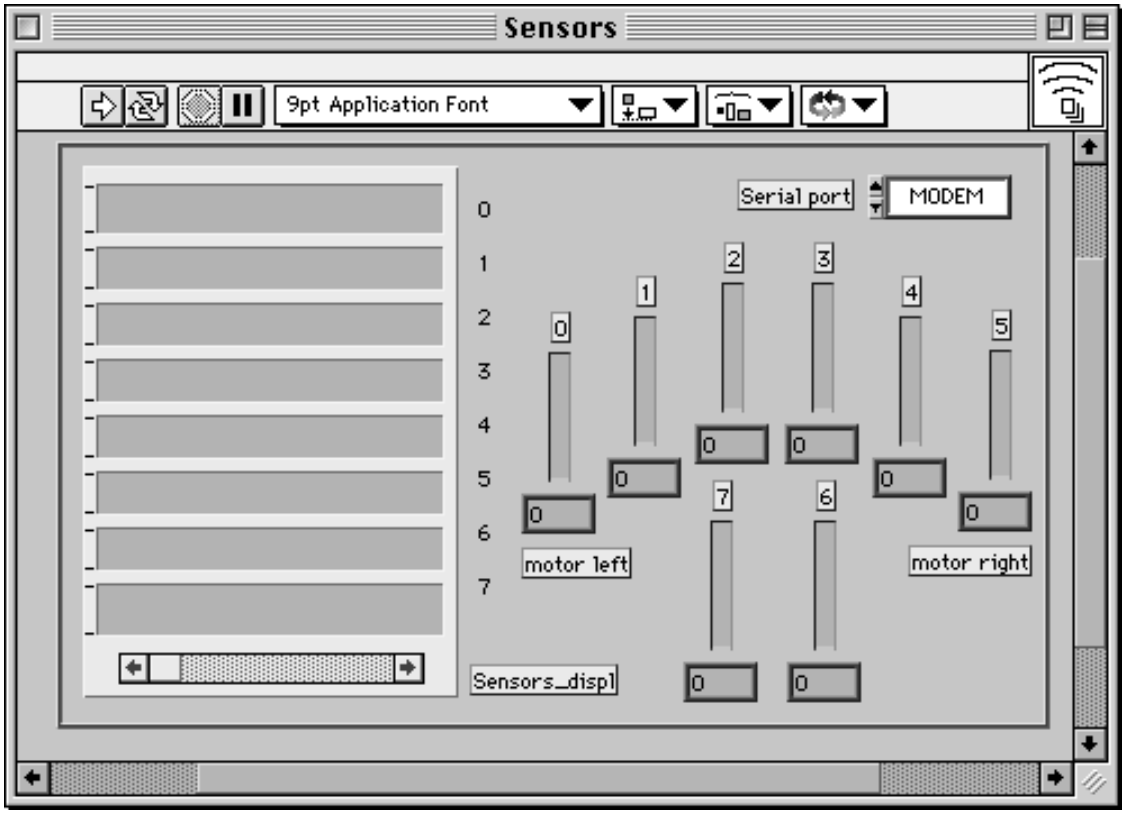


Figure 28: Sensors panel: 8 gauges displaying the infra red values.

Each proximity sensor value is displayed as a gauge. The gauges are placed on the panel like the corresponding sensors on the robot. The exact value received is written underneath. Values are between 0 and 1023. Start the acquisition as before by clicking on the double arrow (to stop the execution, click on the stop icon that will appear). Now you are free to test the response of the sensors. In particular, some materials reflect better than others the infra-red light emitted by the sensors. You are also able to state difference in the individual response of the proximity sensors.

The graph on the left of the panel shows the values for each sensor against time.

7.5 Braitenberg's vehicle

At this stage, you have a good understanding of motors and sensors functionality. In this section we will combine these two modules, as sub-VIs of a main sensory-motor loop VI. Let's open both panels (Motors and Sensors), but do not start them. This way, you will be able to watch, at the same time, data issued from the sensors and those send to the motors. Open the instrument called KheBraitenberg3c. You should see on your screen the panel illustrated in figure 29.

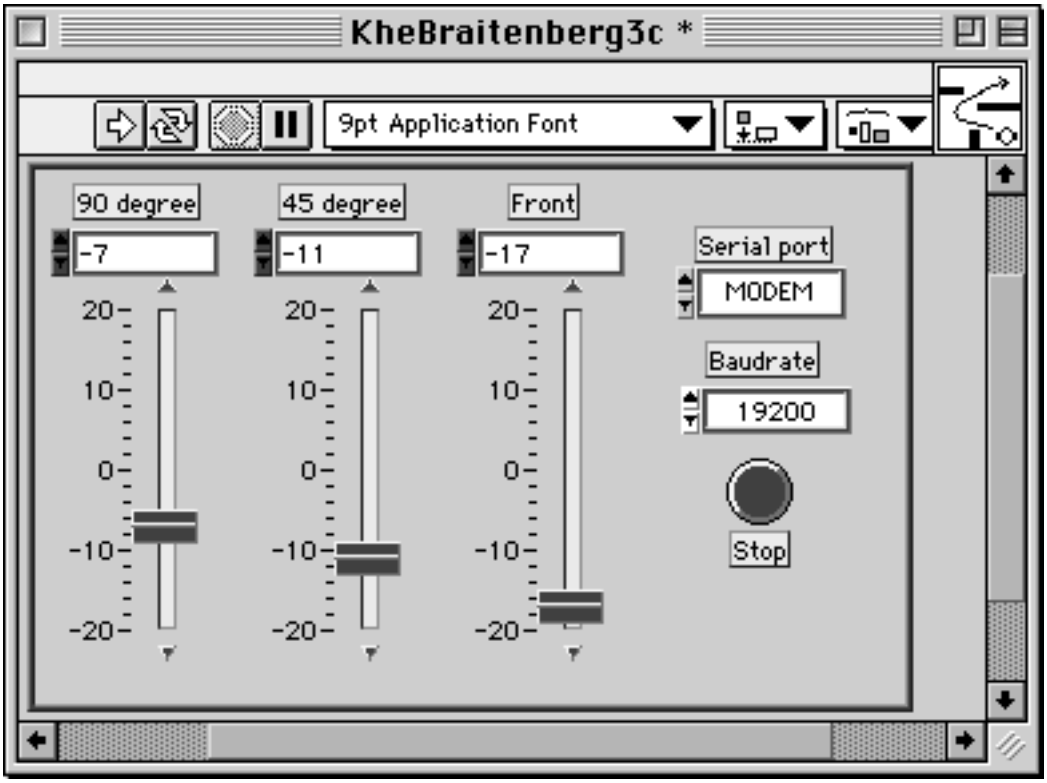


Figure 29: Braitenberg's vehicle panel: 3 sliders defining the link weights.

Each of these three sliders defines the sensibility of the motors reaction to the obstacles of a given group of sensors:

- "Front" corresponds to the two central front sensors (2 and 3),
- "45 degrees" corresponds to the two lateral front sensors (1 and 4),
- "90 degrees" corresponds to the two sensors on the side of the robot (0 and 5).

Set carefully the serial port and the baudrate according to the settings of your Khepera robot. Start the application by clicking on the arrow. It is not necessary to click on the double arrow in the present case. The sensibility can be modified by moving the cursors or writing directly the desired values. Khepera now moves, avoiding bumping into obstacles. In a parallel way you can observe the motor commands and the sensor readings from the "Motors" and "Sensors" panels. Test different sensibilities on its behaviour. This control structure is inspired from the work of V. Braitenberg [Braitenberg84].

Note that this VI uses "Motors" and "Sensors" as sub-VIs. One of the advantages of LabVIEW® is to allow a context-free use of the building modules. This fact is particularly interesting for creating programs in a structured way and for debugging.

The button (inside the panel) labelled "stop" stops the robot and the execution of the VI. It is a much better way than using the stop icon (above rubber), because the stop button stops the robot before stopping the VI.

7.6 Advanced programming

Now that we have executed different manipulations using LabVIEW® and Khepera, it becomes interesting to present how it has been programmed. First, it is important to be able to manipulate LabVIEW® and its rolling menus.

Select the Motors panel and open the diagram using the option ‘Show diagram’ of the menu “Windows”. Your screen now displays the following schema:

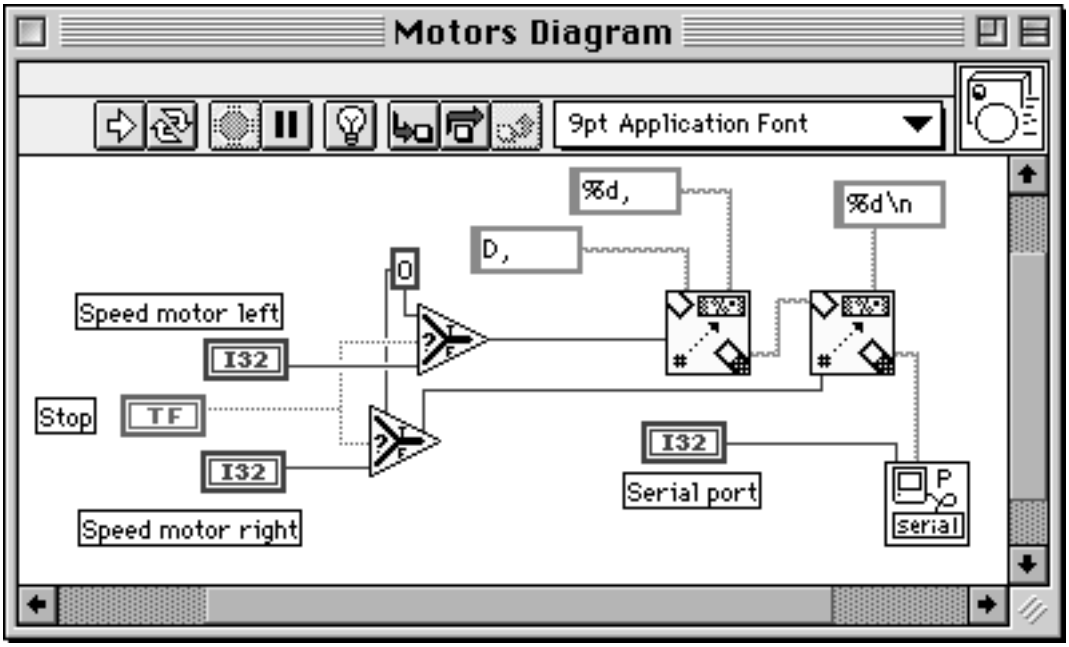


Figure 30: Motors diagram corresponding to the Motors panel.

Each element of the panel used for displaying or getting data corresponds to an icon. So, the box I32 under ‘Speed motor right’ is the getting variable of the corresponding slider on the front panel (of type 32 bits Integer). The same is true for the icon TF, labelled Stop on the left, which is a boolean variable (type True False). This variable allows to stop the motors by sending to each one a null speed value. The triangle represents an indirection controlled by the boolean variable “Stop”: If “Stop” is true, then the output value (on the right) will be 0, if “Stop” is false, then the output value is the Speed variable. These values are formatted by the next two icons to a string of ASCII characters. The character D is placed at the beginning of the string. Then, successively, the two speed values are added and the string is terminated by a carriage return (\n). This string is send to the robot using the serial link.

To control the speed of the wheels from another VI, the “Motors” VI can be used as sub-VI. This use is demonstrated with the KheBraitenberg3c VI. The help window (figure 31) displays positions and semantics associated to the icon.

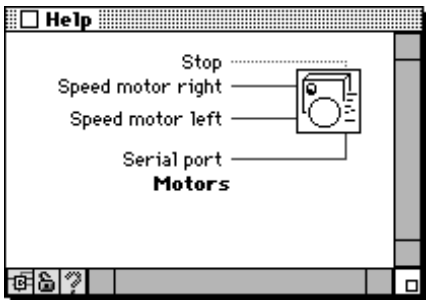


Figure 31: Semantic of the Motors connector.

7.6.1 Sensors

Select the Sensors panel and open the diagram using the option Show diagram of the menu Windows. Your screen now displays the following schema:

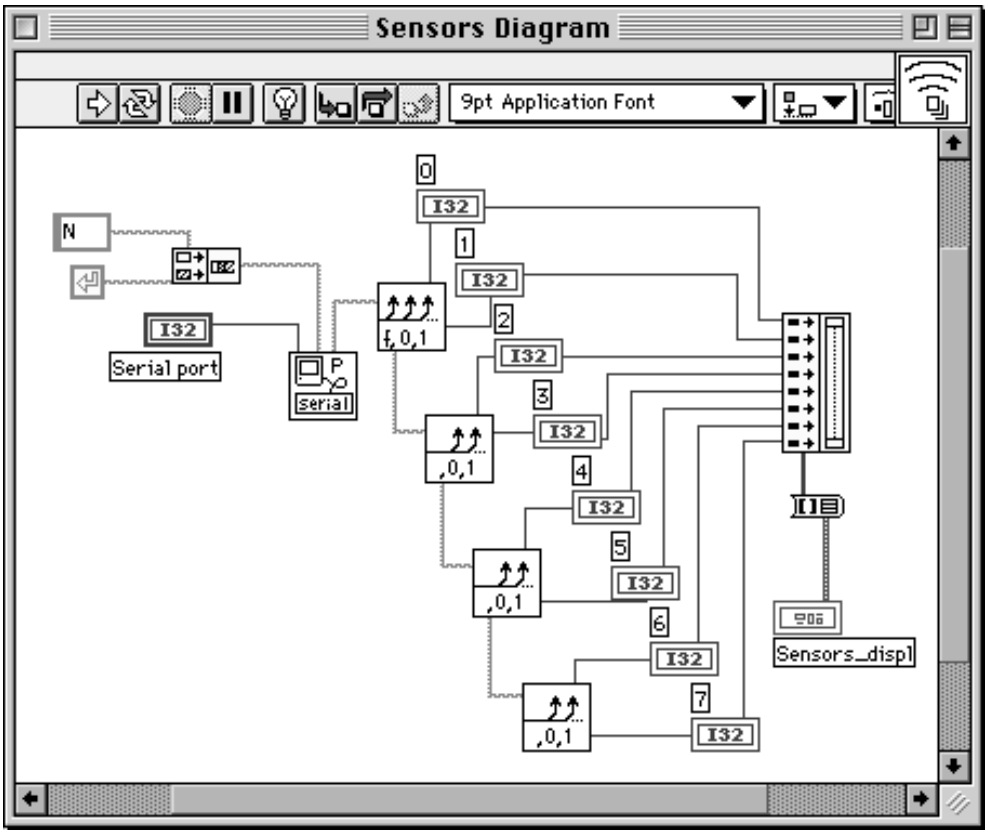


Figure 32: Diagram corresponding to the Sensors panel.

On the contrary to the “Motors” panel which sends values, this panel receives values from the serial link. Eight values are extracted in four steps from the string. These values are put into the variables (type Unsigned 16 bits) corresponding to the panel gauges. They are also put into a vector so to be displayed by Sensors_displ.

These 8 values are transmitted to others modules through the icon used as a con-

nector. This use is demonstrated with the experiment on Braitenberg's vehicle. The help window (figure 33) displays positions and semantics associated to the icon.

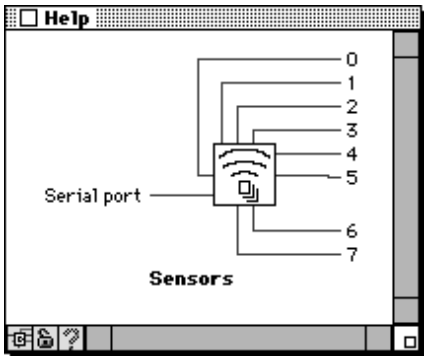


Figure 33: Semantic of the Sensors connector.

7.6.2 Example of Braitenberg's vehicle

Select the Braitenberg panel and open the diagram using the option Show diagram of the menu Windows. Your screen now displays the following schema:

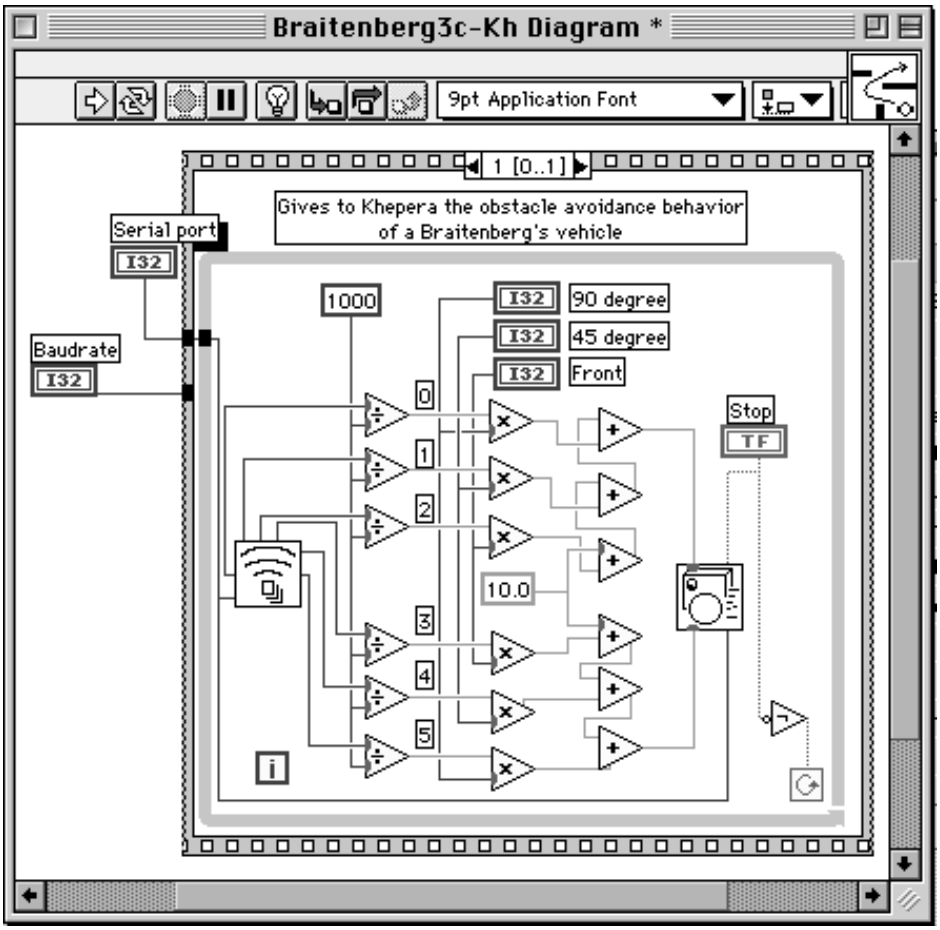


Figure 34: Braitenberg’s vehicle diagram of Braitenberg’s vehicle panel.

This schema may appear complex at first glance. But, as we will see, it takes back elements already studied. It can be decomposed into two parts: serial link initialisation and avoiding behaviour. LabVIEW® is a data-flow controlled language, so the execution order of non-dependent control structures is not fixed. Serial link initialisation and avoidance behaviour are independent; but initialisation must occur first. The sequential structure allows the definition of an execution order, so initialisation is the first element of the sequence. The second element contains the while loop where the avoidance behaviour is executed until the boolean variable Stop becomes True (note the presence of a logical inversion). This boolean variable is also transmitted to the “Motors” VI through its icon. Its action will be to stop both motors. In the same way, the two speeds computed here are sent to the “Motors” VI to be sent through the serial link to Khepera.

Sensors values are received from the “Sensors” icon. They are normalised (/1000) and multiplied by the corresponding sensibility value (type real Single). Front corresponds to the two central sensors (numbers 2 and 3), 45 degree corresponds to the two oblique sensors (numbers 1 and 4) and 90 degree corresponds to the two side sensors (numbers 0 and 5). The result (type single) is added to the value 10,0. The sum of the left sensors (3, 4 and 5) corresponds to motor right (0). The sum of the right sensors (0, 1 and 2) corresponds to motor left (1). These two values are then formatted (I32) and send to the motors. The computation needed is simple and fast enough to control Khepera in real-time without big delays. However, displaying “Motors” and “Sensors” panels is a computational expensive operation. If you want Khepera to move faster, close these panels but don't close the Braitenberg panel!

8 REFERENCES

[Braitenberg84] Braitenberg V., “Vehicles: Experiments in synthetic psychology,” MIT Press, 1984.

[Mondada93b] Mondada F., Franzi E. and Ienne P., “Mobile robot miniaturisation: a tool for investigation in control algorithms.”, ISER3, Kyoto, Japan, 1993.

[National91] National Instruments, LabVIEW manuals for the release 5, january 1998.

APPENDIX A COMMUNICATION PROTOCOL TO CONTROL THE ROBOT

This communication protocol allows complete control of the functionnalities of the robot through a RS232 serial line. The connection configuration needed is presented in section 5.2. The set-up of the serial line of your host computer must correspond to the one set on the robot with the jumpers (running modes 1 to 3). The protocol is constituted by commands and responses, all in standard ASCII codes. A command goes from the host computer to the robot: it is constituted by a capital letter followed, if necessary, by numerical or literal parameters separated by a comma and terminated by a line feed. The response goes from the robot to the host computer: it is constituted by the same letter of the command but in lower case, followed, if necessary, by numerical or literal parameters separated by a comma and terminated by a line feed.

To better understand this protocol we propose a very simple test as following:

- Set the jumpers of the robot for running mode number 1 (See figure 20).
- Set the connection configuration presented in section 5.2.
- Start a terminal emulator on your host computer with the serial line set to 9600 Baud, 8 bit data, 1 start bit, 2 stop bits, no parity.
- Type the capital letter **B** followed by a carriage return or a line feed.
- The robot must respond with **b** followed by an indication of the version of software running on the robot and terminated by a line feed.
- Type the capital letter **N** followed by a carriage return or a line feed.
- The robot must respond with **n** followed by 8 numbers separated by a comma and terminated by a line feed. These numbers are the values of the proximity sensors presents on the robots.
- Retry the same command (N) putting some obstacles on the front of the robot. The response must change.
- Try other commands:

List of Available Commands

(∏ indicates CR (carriage return) **or** LF (line feed). ¶ indicates CR **and** LF.)

A Configure

Format of the command: A, Kp, Ki, Kd∏

Format of the response: a¶

Effect: Set the proportional (Kp), integral (Ki) and derivative (Kd) parameters of the speed controller. At the reset, these parameters are set to standard values: Kp to 3800, Ki to 800, Kd to 100.

B Read software version

Format of the command: B∏

Format of the response: b, version_of_BIOS, version_of_protocol¶

Effect: Give the version of the software present in the EPROM of the robot.

C Set a position to be reached

Format of the command: C,pos_left,pos_right∏

Format of the response: c¶

Effect: Indicate to the wheel position controller an absolute position to be reached. The motion control perform the movement using the three control phases of a trapezoidal speed shape: an acceleration, a constant speed and a deceleration period. These phases are performed according to the parameters selected for the trapezoidal speed controller (command J). The maximum distance that can be given by this command is (2**23)-2 pulses that correspond to 670m. The unit is the pulse that corresponds to 0.08mm. The movement is done immediately after the command is sent. In the case another command is under execution (speed or position control) the last command replaces the precedent one. Any replacement transition follows acceleration and maximal speed constraints.

D Set speed

Format of the command: D, speed_motor_left, speed_motor_right∏

Format of the response: d¶

Effect: Set the speed of the two motors. The unit is the pulse/10 ms that corresponds to 8 millimetres per second. The maximum speed is 127 pulses/ 10ms that correspond to 1m/s.

E Read speed

Format of the command: E∏

Format of the response: e, speed_motor_left, speed_motor_right¶

Effect: Read the instantaneous speed of the two motors. The unit is the pulse/10 ms that corresponds to 8 millimetres per second.

F Configure the position PID controller

Format of the command: F,Kp,Ki,Kd∏

Format of the response: f¶

Effect: Set the proportional (Kp), the integral (Ki) and the derivative (Kd) parameters of the position regulator. At the reset, these parameters are set to standard values: Kp to 3000, Ki to 20, Kd to 4000.

G Set position to the position counter

Format of the command: G, position_motor_left, position_motor_right∏

Format of the response: g¶

Effect: Set the 32 bit position counter of the two motors. The unit is the pulse, that corresponds to 0,08 mm.

H Read position

Format of the command: H∏

Format of the response: h, position_motor_left, position_motor_right¶

Effect: Read the 32 bit position counter of the two motors. The unit is the pulse, that corresponds to 0,08 mm.

I Read A/D input

Format of the command: I, channel_number∏

Format of the response: i, analog_value¶

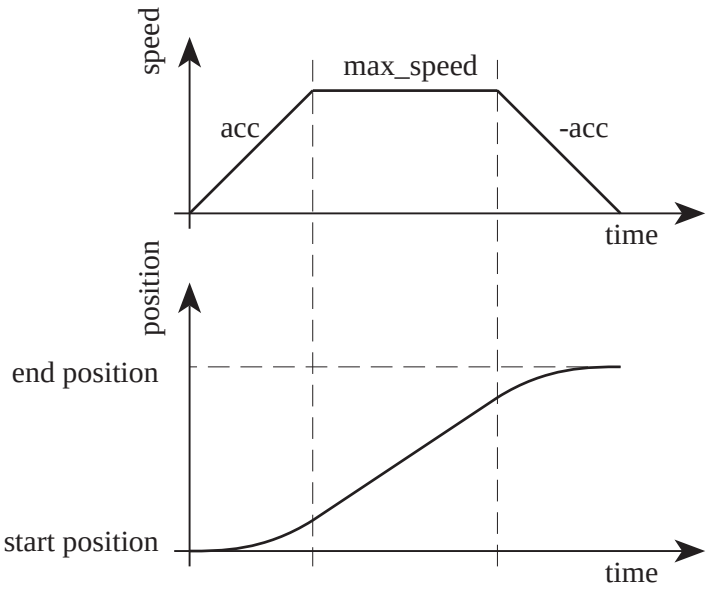
Effect: Read the 10 bit value corresponding to the channel_number analog input. The value 1024 corresponds to an analog value of 4,09 Volts.

J Configure the speed profile controller

Format of the command: J, max_speed_left, acc_left, max_speed_right, acc_right∏

Format of the response: j¶

Effect: Set the speed and the acceleration for the trapezoidal speed shape of the position controller. The max_speed parameter indicates the maximal speed reached during the displacement. The unit for the speed is the pulse/10ms that corresponds to 8 mm/s. The unit for the acceleration is the ((pulse/256)/10 ms)/10 ms, that correspond to 3,125 mm/s^2. At the reset, these parameters are set to standard values: max_speed to 20, acc to 64.



K Read the status of the motion controller

Format of the command: K∏

Format of the response: k, T_left, M_left, E_left, T_right, M_right, E_right¶

Effect: Read the status of the motion controller. The status is given by three numbers for every motor: T (target), M (mode) and E (error). T=0 means that the robot is still on movement. T=1 means that the robot is on the target position. M=0 means that the motor control is in the speed mode. M=1 means that the control is in position mode. M=2 means that the control is in PWM mode. E indicates controller position or speed error.

L Change LED state

Format of the command: L, LED_number, action_number∏

Format of the response: l¶

Effect: Perform an action on one of the two LEDs of the robot. Possible actions are: 0: turn OFF, 1: turn ON, 2: change status. The LED number 0 is the lateral one, the LED number 1 is the frontal one.

N Read proximity sensors

Format of the command: N∏

Format of the response: n,val_sens_left_90°,val_sens_left_45°,val_sens_left_10°, val_sens_right_10°,val_sens_right_45°,val_sens_right_90° ,val_sens_back_right,val_sens_back_left¶

Effect: Read the 10 bit values of the 8 proximity sensors (section 2.1.6.2), from the front sensor situated at the left of the robot, turning clockwise to the back-left sensor.

O Read ambient light sensors

Format of the command: O∏

Format of the response: o,val_sens_left_90°,val_sens_left_45°,val_sens_left_10°, val_sens_right_10°,val_sens_right_45°,val_sens_right_90° ,val_sens_back_right,val_sens_back_left¶

Effect: Read the 10 bit values of the 8 light sensors (section 2.1.6.1), from the front left sensor turning clockwise to the back-left sensor.

P Set PWM (pulse width modulation)

Format of the command: P, pwm_motor_left, pwm_motor_right∏

Format of the response: p¶

Effect: Set the desired PWM amplitude (see “Motors and motor control” on page 6 for more details) on the two motors. The minimum PWM ratio is 0 (0%). The maximal forward ratio (100%) correspond to a value of 255. The maximal backwards ratio (100%) correspond to a value of -255.

T Send a message to an extension turret

Format of the command: T, turret_ID, command∏

Format of the response: t, response¶

Effect: Send a command and return the response of the intelligent extension turret with turret_ID. The list of turrets connected and their ID can be requested with the tool “net”. The command parameter takes the same format as a standard command, including an identification capital letter followed, if necessary, by numerical parameters separated by commas and terminated by a line feed. The response takes the same format, starting with the same letter but in lower case, followed, if necessary, by numerical parameters separated by commas and terminated by a line feed. The command and response formats are specific for every module.

R Read a byte on the extension bus

Format of the command: R, relative_address∏

Format of the response: r, data¶

Effect: Read the data byte available at the relative_address (0...63) of the extension bus.

W Write a byte on the extension bus

Format of the command: W, data, relative_address∏

Format of the response: w¶

Effect: Write the data byte at the relative_address (0...63) of the extension bus.

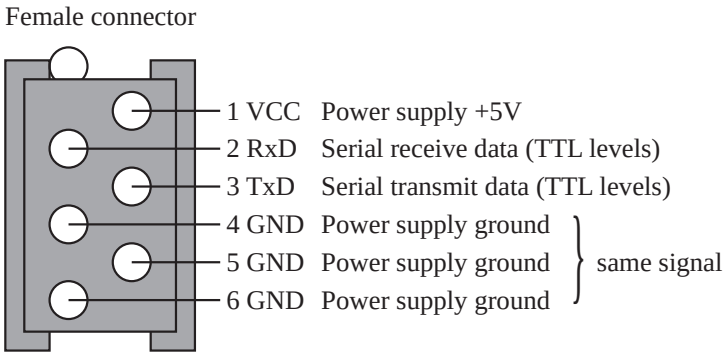


Figure 35: Serial line S connector (RED), placed on the CPU board and on the interface-charger module.

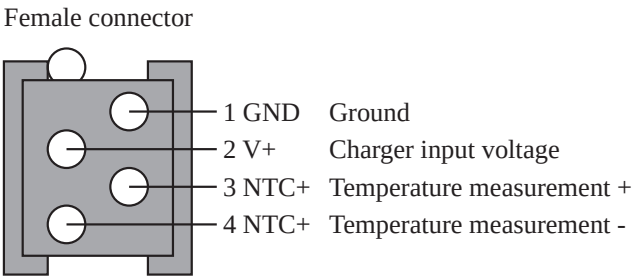
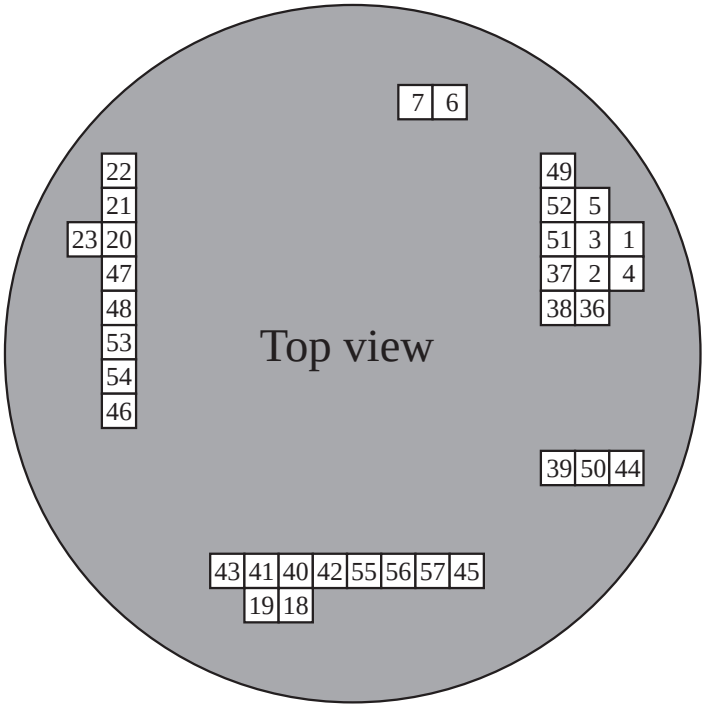


Figure 36: Battery recharger connector, placed on the motor-sensors board and on the charger module.



Pin	Signal	Pin	Signal
1	/Reset	41	D15
2	VCC_Ext	42	A0
3	GND	43	A1
4	VCC	44	A2
5	GND	45	R/W
6	VRef	46	/CSExt
7	GNA	47	F7
18	D8	48	/IRQ6
19	D9	49	MISO
20	D10	50	MOSI
21	D11	51	SCK
22	D12	52	/CSCOM
23	D13	53	TxD
36	CH3	54	RxD
37	CH4	55	A3
38	CH5	56	A4
39	PAI	57	A5
40	D14		

Figure 37: K-extension bus pinning

DB25 female connector

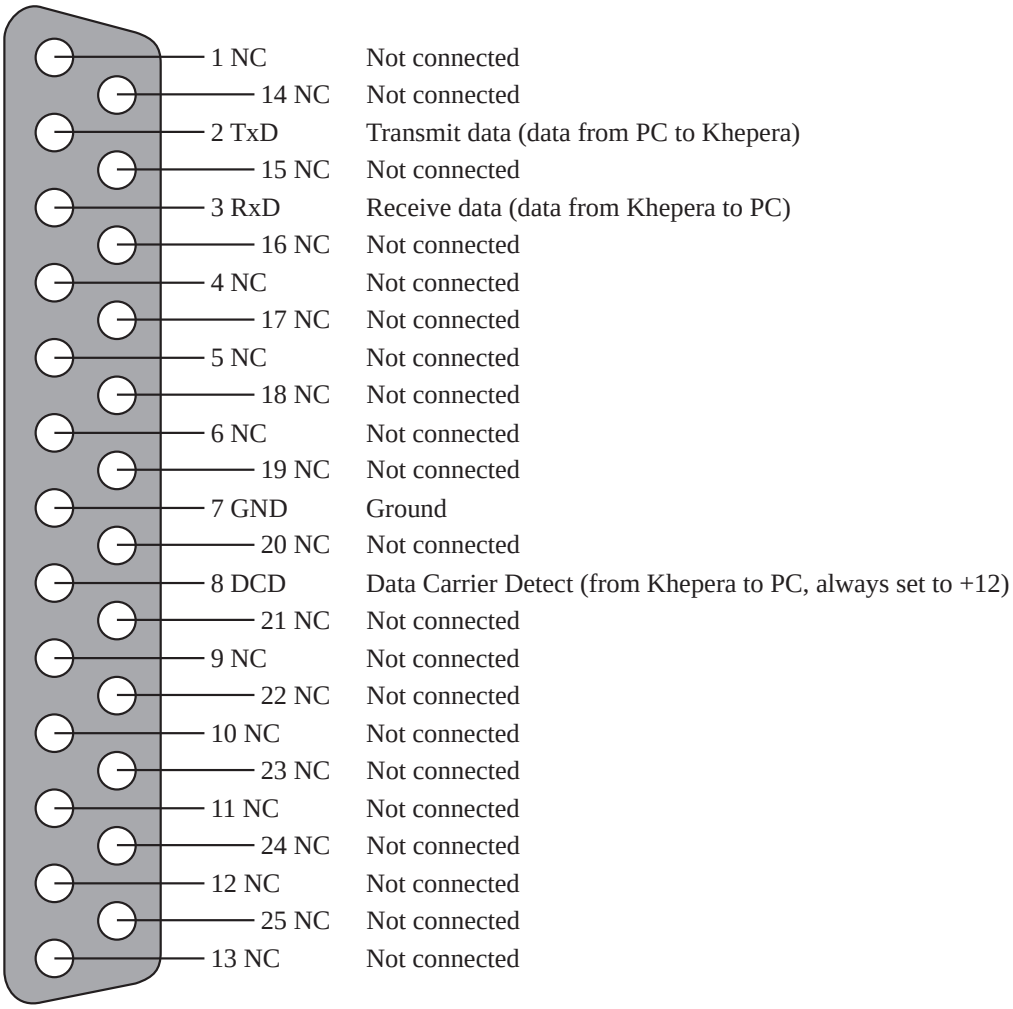


Figure 38: RS232 connector, placed on the interface-charger module.

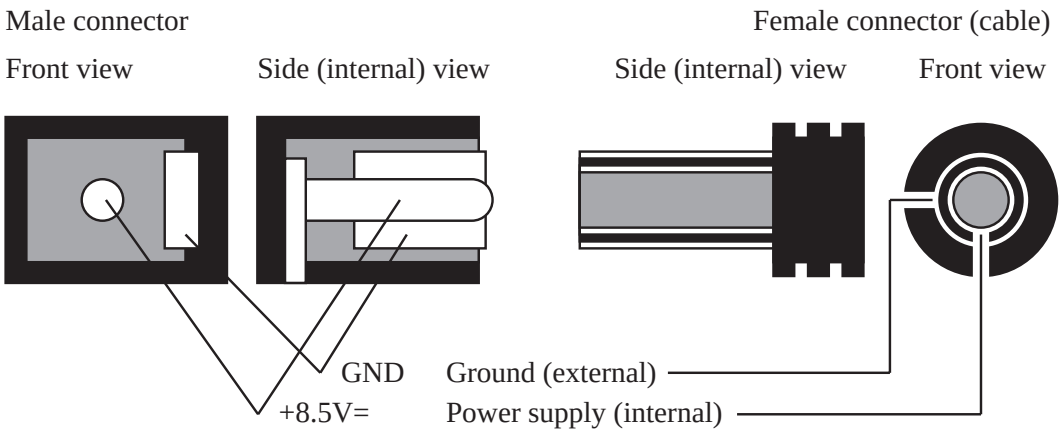


Figure 39: Power supply connector of the interface-charger module.

APPENDIX C HOW TO OPEN THE ROBOT AND CHANGE THE ROM

Changing the ROM of Khepera needs very fine operations. The complete sequence of actions is described in this appendix. **Please follow very carefully these instructions. A wrong action can cause mechanical damages to your Khepera robot. We do assume no responsibility for your wrong manipulations.**

The ROM is situated inside the robot. To change it please:

- Disconnect the robot from the power supply or the battery charger, switch the robot OFF.
- Open the robot by separating the CPU board to the motor-sensors board. Be very careful in this operation and follow the indications given in figure 40 and figure 41.

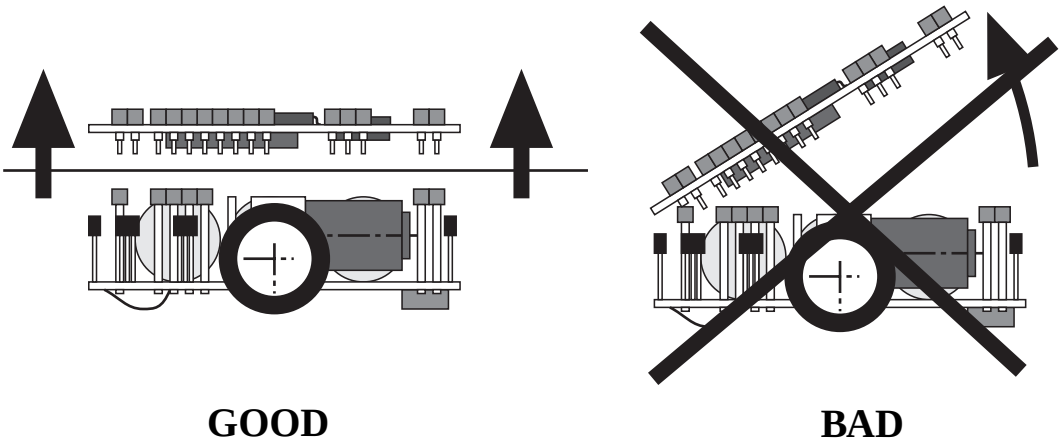


Figure 40: Good and bad way to remove a CPU board from the basic motor and sensor board. To do this operation without damages to the pins, the CPU board must be disconnected carefully and all pins have to be disconnected together. This can be made using a big plastic screwdriver and operating between the CPU board and the white motor blocks. Open some millimetres right, then left, then right.... and be very careful!

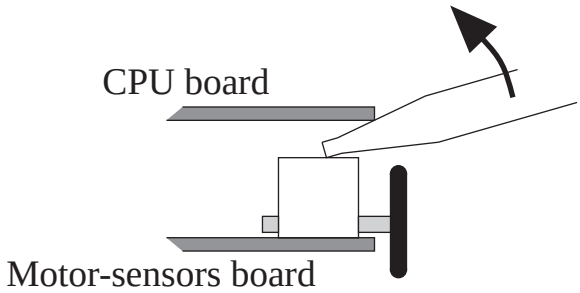


Figure 41: Be very careful opening the robot. The tool should not damage electronic components. The tool should make the effort directly to the green print board or on stable mechanical parts like the white motor boxes indicated in the figure.

- When the CPU and motor board are disconnected, remove the ROM. The ROM is situated on the bottom of the CPU board, as indicated in figure 42.

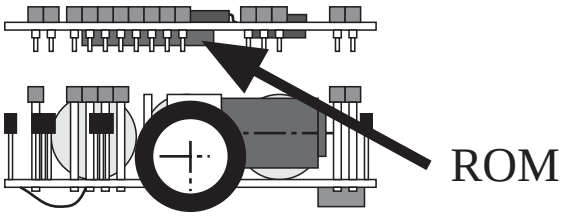


Figure 42: ROM position, on the bottom side of the CPU board.

- To extract the EPROM operate carefully as with the robot. There are two access holes that make the extraction possible as indicated in figure 43.

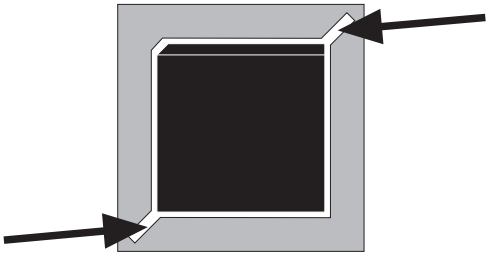


Figure 43: Location of the access holes for the extraction of the ROM.

- Use a specific extraction tool to push out the EPROM, very carefully, and in a parallel way as indicated in figure 44.

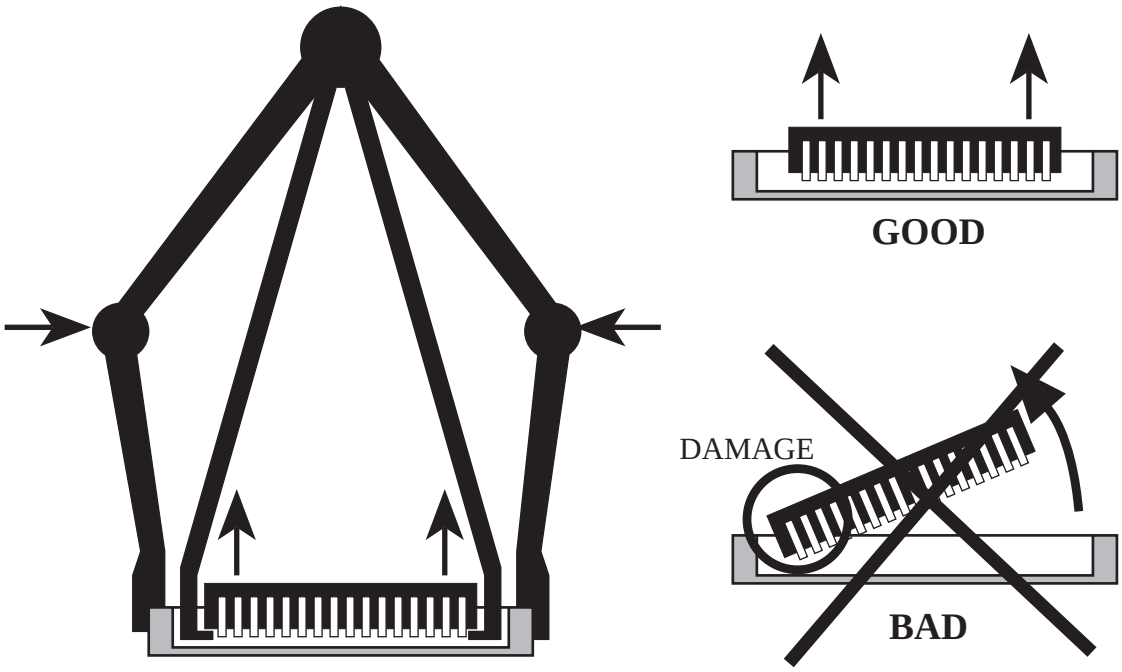


Figure 44: Correct and wrong ROM extraction operation. On the left, the ROM extraction tool.

- Finally plug the new EPROM. Be careful about the orientation, given by a corner of the ROM, as showed in figure 45.

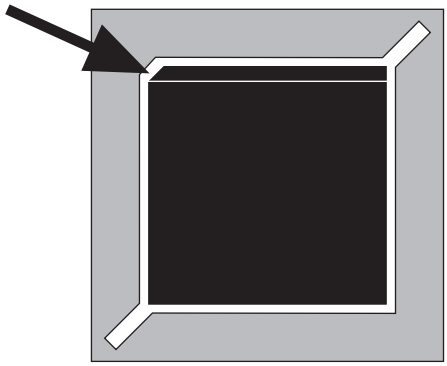


Figure 45: One corner of the chip define its orientation.

- Assemble again CPU and motor-sensors board. Be sure to completely insert the connectors to ensure a good contact.

On PCs, a common terminal emulator is Hyper Terminal. Figure 46 shows the configuration panel set for a communication speed of 9600 baud on COM1.

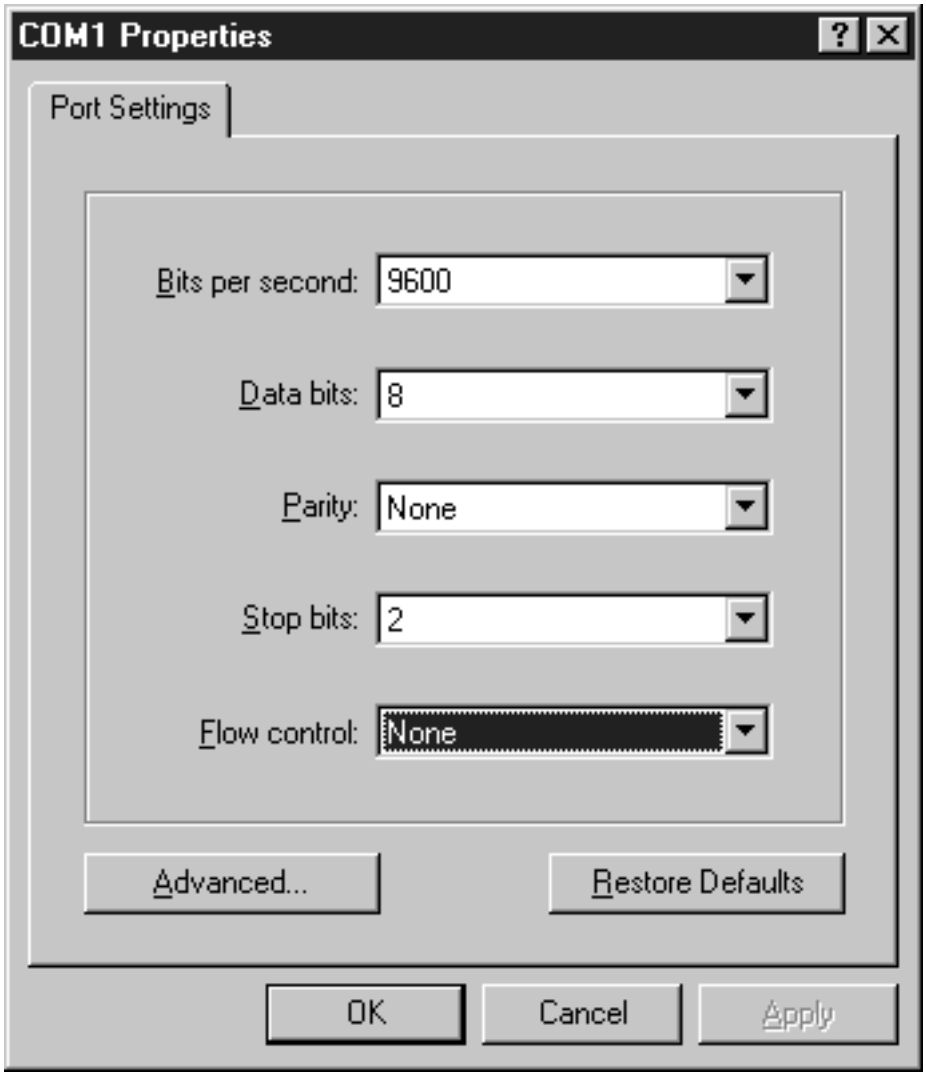


Figure 46: Hyper Terminal configuration panel.

On MAC you can use the software Microphone. The main configuration panel is illustrated in figure 47. You have of course to choose the correct “Connection Port” where the Khepera is connected.

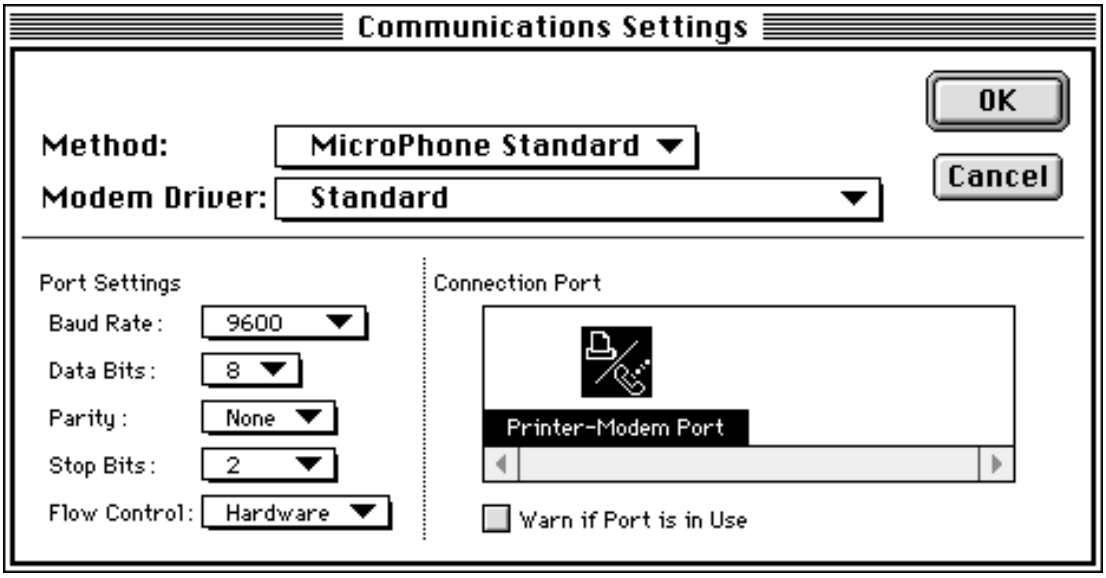


Figure 47: Microphone communications settings.

On LINUX you can use MINICOM, where the settings can be adjusted running “minicom -s” as root or by changing the settings with the CTRL-A O command. The correct settings are illustrated in figure 48 and figure 49.

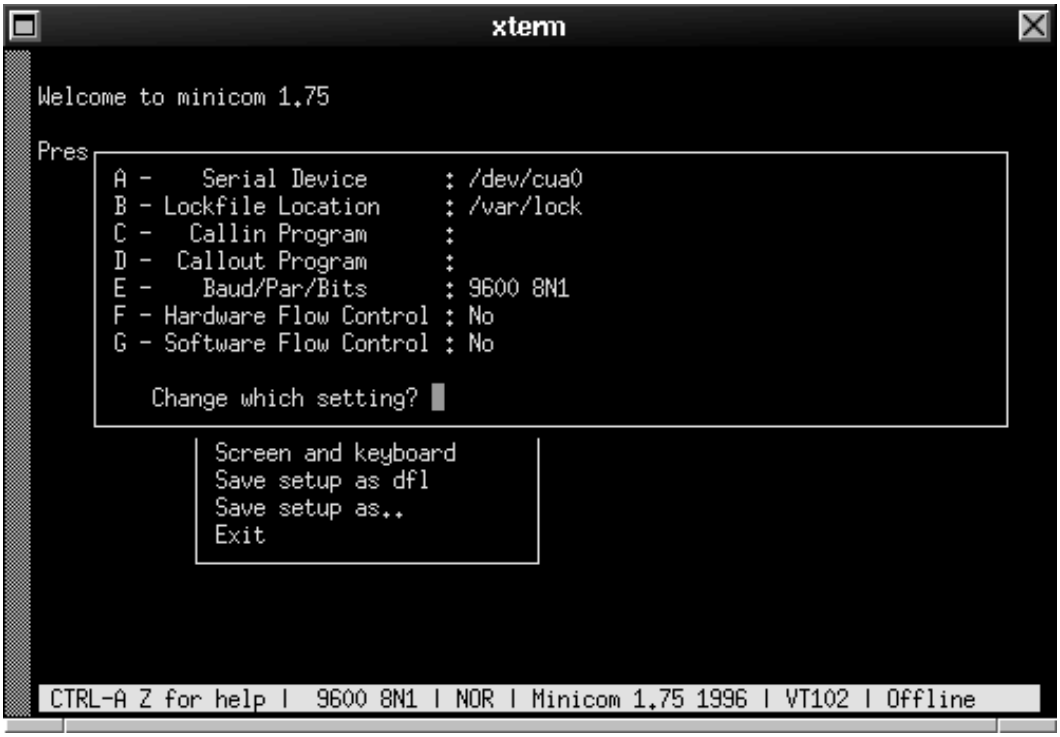


Figure 48: Serial port setting in MINICOM.

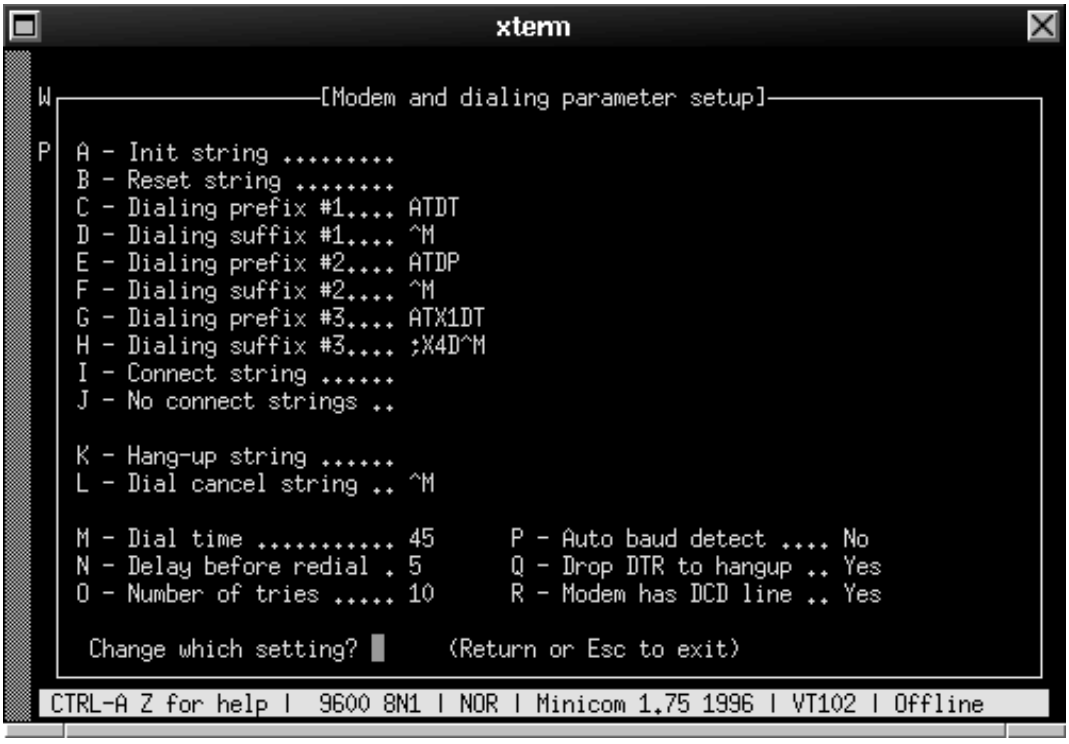
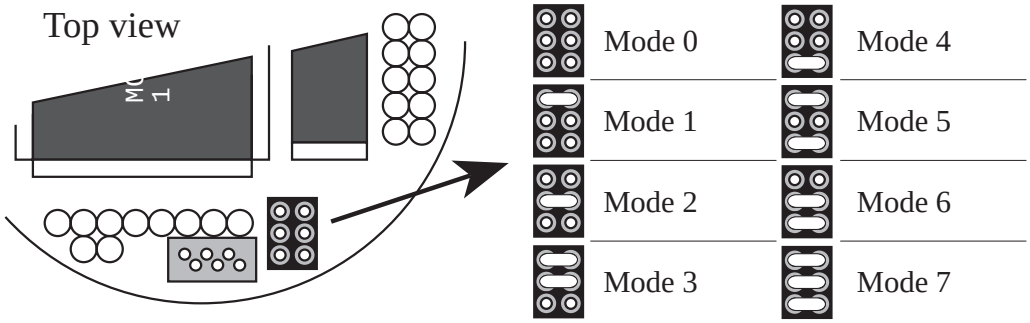


Figure 49: Modem settings in MINICOM.

On SUN, a common tool is “tip”, used on the /dev/ttya or /dev/ttyb ports. More details are given in the manual “man tip”.



Running modes (see “Jumpers, reset button and settings” on page 5):

0. **Demonstration mode**: execution of a Braitenberg vehicle algorithm (number 3 according to the “Vehicle” book [Braitenberg84]) for obstacle avoidance.
1. Mode for the control of the robot by the **serial communication protocol** using a serial link with a communication speed of **9600 Baud**.
2. Mode for the control of the robot by the **serial communication protocol** using a serial link with a communication speed of **19200 Baud**.
3. Mode for the control of the robot by the **serial communication protocol** using a serial link with a communication speed of **38400 Baud**.
4. **User application mode**: start an application stored into the EPROM (if any).
5. **Downloading mode**: in this mode the robot waits for a program to be transferred (**S format, 9600 Baud**).
6. **Downloading mode**: in this mode the robot waits for a program to be transferred (**S format, 38400 Baud**).
7. **Test** of the functionality of the robot. The result of successive tests is given on the serial link (**9600 Baud**).

The serial link set-up is always 8 bit, 1 start bit, 2 stop bit, no parity. Only the baud rate changes. The set-up of the jumpers can be changed at any time. **If the robot is running it is necessary to reset it to make the set-up effective.**